

[illegible]

```

AAAAAA  SSSSSSSS  YY  YY  NN  NN  CCCCCCCC  HH  HH  RRRRRRRR  000000  NN  NN
AAAAAA  SSSSSSSS  YY  YY  NN  NN  CCCCCCCC  HH  HH  RRRRRRRR  000000  NN  NN
AA  AA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AA  AA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AA  AA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AA  AA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AA  AA  SSSSSS  YY  YY  NN  NN  CC  HHHHHHHHHH  RRRRRRRR  00  00  NN  NN
AA  AA  SSSSSS  YY  YY  NN  NN  CC  HHHHHHHHHH  RRRRRRRR  00  00  NN  NN
AAAAAAAAA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AAAAAAAAA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AA  AA  SS  YY  YY  NN  NN  CC  HH  HH  RR  RR  00  00  NN  NN
AA  AA  SSSSSSSS  YY  YY  NN  NN  CCCCCCCC  HH  HH  RR  RR  000000  NN  NN
AA  AA  SSSSSSSS  YY  YY  NN  NN  CCCCCCCC  HH  HH  RR  RR  000000  NN  NN

```

  

```

LL  IIIIII  SSSSSSSS
LL  IIIIII  SSSSSSSS
LL  II  SS
LL  II  SS
LL  II  SS
LL  II  SSSSSS
LL  II  SSSSSS
LL  II  SS
LL  II  SS
LL  II  SS
LLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLL  IIIIII  SSSSSSSS

```

```
1 0001 0 MODULE ASYNCHRON(%TITLE 'Asynchronous service management'
2 0002 0 IDENT = 'V04-002'
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1
7 0007 1 *****
8 0008 1 *
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
11 0011 1 * ALL RIGHTS RESERVED.
12 0012 1 *
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
18 0018 1 * TRANSFERRED.
19 0019 1 *
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
22 0022 1 * CORPORATION.
23 0023 1 *
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
26 0026 1 *
27 0027 1 *
28 0028 1 *****
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY:
33 0033 1 Job controller.
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1 This module contains the routines that manage services that complete
37 0037 1 asynchronously to the original request. Many such instances require
38 0038 1 communication with remote job controllers in a cluster.
39 0039 1
40 0040 1 ENVIRONMENT:
41 0041 1 VAX/VMS user and kernel mode.
42 0042 1 --
43 0043 1
44 0044 1 AUTHOR: M. Jack, CREATION DATE: 16-Feb-1982
45 0045 1
46 0046 1 MODIFIED BY:
47 0047 1
48 0048 1 V04-002 JAK0236 J A Krycka 14-Sep-1984
49 0049 1 Collect more diagnostic information.
50 0050 1
51 0051 1 V04-001 JAK0235 J A Krycka 12-Sep-1984
52 0052 1 Detect and repair a corrupted incomplete services list in
53 0053 1 SCAN_INCOMPLETE_SERVICES.
54 0054 1
55 0055 1 V03-011 JAK0224 J A Krycka 24-Aug-1984
56 0056 1 In ENTER_REMOTE_REQUEST set a flag if there is no doorbell lock
57 0057 1 defined for the remote job controller (indicating that the
```

```
: 58      0058 1 | remote node is not in the cluster or the remote job controller
: 59      0059 1 | does not have the queue file open).
: 60      0060 1 |
: 61      0061 1 | V03-010 KPL0003      P Lieberwirth, 30-Jul-1984
: 62      0062 1 | Fix ALL bugs introduced in V03-008.
: 63      0063 1 |
: 64      0064 1 | V03-009 KPL0002      P Lieberwirth, 30-Jul-1984
: 65      0065 1 | Sigh, fix bug in V03-008. When rewriting predecessor,
: 66      0066 1 | do not update predecessor pointer to be the deallocated
: 67      0067 1 | record. (That took someone with the brain of a turnip.)
: 68      0068 1 |
: 69      0069 1 | V03-008 KPL0001      P Lieberwirth, 19-Jul-1984
: 70      0070 1 | Rewrite predecessor before deallocating SRQ in routine
: 71      0071 1 | SCAN_INCOMPLETE_SERVICES. This avoids corrupting the
: 72      0072 1 | incomplete service list at the possible cost of losing
: 73      0073 1 | a deallocated record and extra I/Os.
: 74      0074 1 |
: 75      0075 1 | V03-007 JAK0213      J A Krycka      18-May-1984
: 76      0076 1 | Continuation of V03-006. Use newly created LCK$M_NODLCKBLK
: 77      0077 1 | (no deadlock on blocking AST) option on enqueue instead.
: 78      0078 1 |
: 79      0079 1 | V03-006 JAK0208      J A Krycka      08-May-1984
: 80      0080 1 | Use LCK$M_NODLCKWT (no deadlock wait) option on enqueue service
: 81      0081 1 | for the remote doorbell lock to avoid having the lock manager
: 82      0082 1 | declare a deadlock situation.
: 83      0083 1 |
: 84      0084 1 | V03-005 GRR0001      Gregory R. Robert      09-Sep-1983
: 85      0085 1 | Supply missing dot in call to delete_files.
: 86      0086 1 |
: 87      0087 1 | V03-004 MLJ0115      Martin L. Jack, 30-Jul-1983
: 88      0088 1 | Changes for job controller baselevel.
: 89      0089 1 |
: 90      0090 1 | V03-003 MLJ0114      Martin L. Jack, 23-Jun-1983
: 91      0091 1 | Changes for job controller baselevel.
: 92      0092 1 |
: 93      0093 1 | V03-002 MLJ0113      Martin L. Jack, 26-May-1983
: 94      0094 1 | Changes for job controller baselevel.
: 95      0095 1 |
: 96      0096 1 | V03-001 MLJ0112      Martin L. Jack, 29-Apr-1983
: 97      0097 1 | Changes for job controller baselevel.
: 98      0098 1 |
: 99      0099 1 | **
```

```

101 0100 1 REQUIRE 'SRC$:JOBCTLDEF';
102 1141 1
103 1142 1
104 1143 1 FORWARD ROUTINE
105 1144 1 CREATE SRQ RECORD,
106 1145 1 PROCESS_REMOTE_SERVICES: L OUTPUT_1,
107 1146 1 SCAN_INCOMPLETE_SERVICES: NOVALUE,
108 1147 1 REMOTE_BLOCKING_AST: NOVALUE,
109 1148 1 REMOTE_COMPLETION_NONAST: NOVALUE,
110 1149 1 REMOTE_COMPLETION_AST: NOVALUE,
111 1150 1 ENTER_REMOTE_REQUEST: NOVALUE,
112 1151 1 ENTER_REMOTE_REQUEST_AST: NOVALUE,
113 1152 1 QUEUE_MASTER_AST: NOVALUE;
114 1153 1
115 1154 1
116 1155 1 EXTERNAL ROUTINE
117 1156 1 ABORT_EXECUTION,
118 1157 1 AFTER_AST: NOVALUE,
119 1158 1 ALLOCATE_MEMORY,
120 1159 1 ALLOCATE_RECORD: L OUTPUT_2,
121 1160 1 BROADCAST_MESSAGE: NOVALUE,
122 1161 1 COMPLETE_JOB: NOVALUE,
123 1162 1 CREATE_SRB: NOVALUE,
124 1163 1 DEALLOCATE_MEMORY: NOVALUE,
125 1164 1 DEALLOCATE_RECORD: NOVALUE,
126 1165 1 DELETE_FILES: NOVALUE,
127 1166 1 FIND_PENDING_JOBS: NOVALUE,
128 1167 1 FLUSH_RECORD: NOVALUE,
129 1168 1 LOCK_QUEUE_FILE: NOVALUE,
130 1169 1 PAUSE_EXECUTION,
131 1170 1 READ_RECORD,
132 1171 1 RELEASE_RECORD: NOVALUE,
133 1172 1 RESET_EXECUTOR_QUEUE: NOVALUE,
134 1173 1 RESUME_EXECUTION,
135 1174 1 REWRITE_RECORD: NOVALUE,
136 1175 1 SCHEDULE_NONAST: NOVALUE,
137 1176 1 SEND_SERVICE_RESPONSE_MESSAGE: NOVALUE,
138 1177 1 START_EXECUTION,
139 1178 1 START_SYMBIONT_STREAM,
140 1179 1 STOP_SYMBIONT_STREAM,
141 1180 1 UNLOCK_QUEUE_FILE: NOVALUE,
142 1181 1 UPDATE_GETOUT_DATA: NOVALUE;
143 1182 1
144 1183 1
145 1184 1 LITERAL
146 1185 1 K_COMPLETE= 0, ! Complete request with status
147 1186 1 K_DEALLOCATE= 1, ! Deallocate request
148 1187 1 K_RELEASE= 2, ! Leave request in queue
149 1188 1 K_REWRITE= 3, ! Leave request in queue and rewrite
150 1189 1
151 1190 1
152 1191 1 BUILTIN
153 1192 1 TESTBITSC,
154 1193 1 TESTBITSS;

```

```
156 1194 1 GLOBAL ROUTINE CREATE_SRQ_RECORD(FUNC,P1,P2,P3,P4,P5,P6,P7)=
157 1195 1
158 1196 1 ++
159 1197 1
160 1198 1 FUNCTIONAL DESCRIPTION:
161 1199 1 This routine allocates, initializes, and enqueues an incomplete service
162 1200 1 record.
163 1201 1
164 1202 1 INPUT PARAMETERS:
165 1203 1 FUNC - Function code.
166 1204 1 P1-P7 - Function-specific parameters.
167 1205 1
168 1206 1 IMPLICIT INPUTS:
169 1207 1 NONE
170 1208 1
171 1209 1 OUTPUT PARAMETERS:
172 1210 1 NONE
173 1211 1
174 1212 1 IMPLICIT OUTPUTS:
175 1213 1 NONE
176 1214 1
177 1215 1 ROUTINE VALUE:
178 1216 1 Completion status.
179 1217 1
180 1218 1 SIDE EFFECTS:
181 1219 1 NONE
182 1220 1
183 1221 1 --
184 1222 1
185 1223 2 BEGIN
186 1224 2 LOCAL
187 1225 2 SQH: REF BBLOCK, ! Pointer to SQH
188 1226 2 SRQ_N, ! Record number of SRQ record
189 1227 2 SRQ: REF BBLOCK, ! Pointer to SRQ record
190 1228 2 STATUS; ! Status return
191 1229 2
192 1230 2
193 1231 2 ! Allocate the queue record, and return if no more.
194 1232 2
195 1233 2 STATUS = ALLOCATE_RECORD( ; SRQ_N, SRQ);
196 1234 2 IF NOT .STATUS THEN RETURN .STATUS;
197 1235 2
198 1236 2
199 1237 2 ! Initialize the incomplete service record.
200 1238 2
201 1239 2 SRQ[SYMSB_TYPE] = SYMSK_SRQ;
202 1240 2 SRQ[SRQ$L_FUNCTION_CODE] = .FUNC;
203 1241 2 COPY_SYSID(THIS_SYSID, SRQ[SRQ$I_SENDING_SYSID]);
204 1242 2
205 1243 2
206 1244 2 CASE .FUNC FROM SRQ$K_START_JOB TO SRQ$K_START_SYMBIONT OF
207 1245 2 SET
208 1246 2
209 1247 2
210 1248 2 [INRANGE, OTRANGE]:
211 1249 2 0;
212 1250 2
```

```
213 1251 2
214 1252
215 1253 [SRQ$K_START_JOB]:
216 1254 BEGIN
217 1255 BIND
218 1256 SMQ_N = P1, ! Record number of SMQ
219 1257 SMQ = P2: REF BBLOCK, ! Pointer to SMQ
220 1258 SJH_N = P3, ! Record number of SJH
221 1259 SJH = P4: REF BBLOCK; ! Pointer to SJH
222 1260
223 1261 SRQ[SRQ$V_NO_RESPONSE] = TRUE;
224 1262 SJH[SJH$V_STARTING] = TRUE;
225 1263 COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
226 1264 SRQ[SRQ$L_P1] = .SMQ_N;
227 1265 SRQ[SRQ$L_P2] = .SJH_N;
228 1266 END;
229 1267
230 1268 [SRQ$K_ABORT_JOB]:
231 1269 BEGIN
232 1270 BIND
233 1271 SMQ_N = P1, ! Record number of SMQ
234 1272 SMQ = P2: REF BBLOCK, ! Pointer to SMQ
235 1273 SJH_N = P3, ! Record number of SJH
236 1274 SJH = P4: REF BBLOCK; ! Pointer to SJH
237 1275
238 1276 SJH[SJH$V_ABORTING] = TRUE;
239 1277 COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
240 1278 SRQ[SRQ$L_P1] = .SMQ_N;
241 1279 SRQ[SRQ$L_P2] = .SJH_N;
242 1280 END;
243 1281
244 1282 [SRQ$K_SYNCHRONIZE_JOB]:
245 1283 BEGIN
246 1284 BIND
247 1285
248 1286 SJH_N = P1, ! Record number of SJH
249 1287 SJH = P2: REF BBLOCK; ! Pointer to SJH
250 1288
251 1289 SJH[SJH$V_SYNCHRONIZE] = TRUE;
252 1290 SRQ[SRQ$V_STALLED] = TRUE;
253 1291 COPY SYSID(THIS_SYSID, SRQ[SRQ$T_RECEIVING_SYSID]);
254 1292 SRQ[SRQ$L_P1] = .SJH_N;
255 1293 END;
256 1294
257 1295 [SRQ$K_START_QUEUE]:
258 1296 BEGIN
259 1297 BIND
260 1298
261 1299 SMQ_N = P1, ! Record number of SMQ
262 1300 SMQ = P2: REF BBLOCK; ! Pointer to SMQ
263 1301
264 1302 SMQ[SMQ$V_STARTING] = TRUE;
265 1303 SMQ[SMQ$V_STOPPED] = SMQ[SMQ$V_PAUSED] = FALSE;
266 1304 COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
267 1305 SRQ[SRQ$L_P1] = .SMQ_N;
268 1306 END;
269 1307 2
```

```

270 1308
271 1309
272 1310
273 1311
274 1312
275 1313
276 1314
277 1315
278 1316
279 1317
280 1318
281 1319
282 1320
283 1321
284 1322
285 1323
286 1324
287 1325
288 1326
289 1327
290 1328
291 1329
292 1330
293 1331
294 1332
295 1333
296 1334
297 1335
298 1336
299 1337
300 1338
301 1339
302 1340
303 1341
304 1342
305 1343
306 1344
307 1345
308 1346
309 1347
310 1348
311 1349
312 1350
313 1351
314 1352
315 1353
316 1354
317 1355
318 1356
319 1357
320 1358
321 1359
322 1360
323 1361
324 1362
325 1363
326 1364

```

```

[SRQ$K_STOP_QUEUE]:
BEGIN
BIND
    SMQ_N      = P1,
    SMQ-       = P2:  REF BBLOCK;
    ! Record number of SMQ
    ! Pointer to SMQ

    SMQ[SMQ$V_STOPPING] = TRUE;
    COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
    SRQ[SRQ$L_P1] = .SMQ_N;
END;

[SRQ$K_PAUSE_QUEUE]:
BEGIN
BIND
    SMQ_N      = P1,
    SMQ-       = P2:  REF BBLOCK;
    ! Record number of SMQ
    ! Pointer to SMQ

    SMQ[SMQ$V_PAUSING] = TRUE;
    COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
    SRQ[SRQ$L_P1] = .SMQ_N;
END;

[SRQ$K_RESUME_QUEUE]:
BEGIN
BIND
    SMQ_N      = P1,
    SMQ-       = P2:  REF BBLOCK,
    FLAGS      = P3:  BBLOCK,
    ALIGNMENT  = P4,
    RELATIVE   = P5,
    SEARCH_LEN  = P6,
    SEARCH_ADDR = P7;
    ! Record number of SMQ
    ! Pointer to SMQ
    ! Resume control flags
    ! Alignment pages
    ! Relative page offset
    ! Search string length
    ! Search string address

    SMQ[SMQ$V_RESUMING] = TRUE;
    COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
    SRQ[SRQ$L_P1] = .SMQ_N;
    SRQ[SRQ$L_P2] = .FLAGS;
    SRQ[SRQ$L_P3] = .ALIGNMENT;
    SRQ[SRQ$L_P4] = .RELATIVE;
    CH$WCHAR(.SEARCH_LEN, SRQ[SRQ$T_P5]);
    CH$MOVE(.SEARCH_LEN, .SEARCH_ADDR, SRQ[SRQ$T_P5]+1);
END;

[SRQ$K_RESET_QUEUE]:
BEGIN
BIND
    SMQ_N      = P1,
    SMQ-       = P2:  REF BBLOCK;
    ! Record number of SMQ
    ! Pointer to SMQ

    SMQ[SMQ$V_RESETTING] = TRUE;
    COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
    SRQ[SRQ$L_P1] = .SMQ_N;
END;

```

```

327 1365 2
328 1366
329 1367 [SRQ$K_BROADCAST_MESSAGE]:
330 1368 BEGIN
331 1369 BIND
332 1370     SYSID      = P1,
333 1371     USERNAME   = P2:  REF VECTOR[.BYTE],
334 1372     LENGTH     = P3,
335 1373     ADDRESS    = P4;
336 1374
337 1375     SRQ[SRQ$V_NO_RESPONSE] = TRUE;
338 1376     COPY SYSID(.SYSID, SRQ[SRQ$T_RECEIVING_SYSID]);
339 1377     CH$MOVE(SRQ$S_BRDCST_USERNAME, .USERNAME, SRQ[SRQ$T_BRDCST_USERNAME]);
340 1378     SRQ[SRQ$W_BRDCST_LENGTH] = .LENGTH;
341 1379     CH$MOVE(.LENGTH, .ADDRESS, SRQ[SRQ$T_BRDCST_TEXT]);
342 1380 END;
343 1381
344 1382 [SRQ$K_DELETE_FILES]:
345 1383 BEGIN
346 1384 BIND
347 1385     SJH      = P1:  REF BBLOCK,      ! Pointer to SJH
348 1386     SQR_N   = P2:                      ! Record number of SQR
349 1387
350 1388
351 1389     SRQ[SRQ$V_NO_RESPONSE] = TRUE;
352 1390     COPY SYSID(SJH[SJH$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
353 1391     SRQ[SRQ$L_P1] = .SQR_N;
354 1392 END;
355 1393
356 1394 [SRQ$K_START_SYMBIONT]:
357 1395 BEGIN
358 1396 BIND
359 1397     SMQ_N   = P1,
360 1398     SMQ     = P2:  REF BBLOCK;      ! Record number of SMQ
361 1399                                     ! Pointer to SMQ
362 1400
363 1401     SRQ[SRQ$V_STALLED] = TRUE;
364 1402     COPY SYSID(SMQ[SMQ$T_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
365 1403     SRQ[SRQ$L_P1] = .SMQ_N;
366 1404 END;
367 1405
368 1406 TES;
369 1407
370 1408
371 1409 IF NOT .SRQ[SRQ$V_NO_RESPONSE]
372 1410 THEN
373 1411     CREATE_SRB(SRQ[SRQ$T_SRB]);
374 1412
375 1413
376 1414 ! If services of another job controller are required, signal it.
377 1415
378 1416
379 1417 IF SYSID_NEQ(THIS_SYSID, SRQ[SRQ$T_RECEIVING_SYSID])
380 1418 AND NOT .SRQ[SRQ$V_STALLED]
381 1419 THEN
382 1420     ENTER_REMOTE_REQUEST(SRQ[SRQ$T_RECEIVING_SYSID]);
383 1421

```

```

: 384      1422  2
: 385      1423  2 ! Enqueue the record to the incomplete service list.
: 386      1424  2
: 387      1425  2 SQH = READ_RECORD(SQH$K_RECNO);
: 388      1426  2 SRQ[SYMSL_LINK] = .SQH[SQH$LINK_INCOMPLETE_SERVICE_LIST];
: 389      1427  2 SQH[SQH$LINK_INCOMPLETE_SERVICE_LIST] = .SRQ_N;
: 390      1428  2 REWRITE_RECORD(.SRQ_N);
: 391      1429  2 REWRITE_RECORD(SQH$K_RECNO);
: 392      1430  2
: 393      1431  2
: 394      1432  2 ! Return 0 to indicate that the service is incomplete.
: 395      1433  2
: 396      1434  2 0
: 397      1435  1 END;

```

```

.TITLE ASYNCHRON Asynchronous service management
.IDENT \V04-002\

```

```

.PSECT COMMON,NOEXE, OVR,2

```

```

00000 DIAG_STORAGE_BASE:
      .BLKB 0
00000 DIAG_TRACE:
      .BLKB 96
00060 DIAG_COUNT:
      .BLKB 96
000C0 DIAG_FLAGS:
      .BLKB 4
000C4 WORK_AREA:
      .BLKB 44
000F0 SNDJBC_COUNT:
      .BLKB 132
00174 GETQUI_COUNT:
      .BLKB 40
0019C SNDACC_COUNT:
      .BLKB 28
001B8 SNDSMB_COUNT:
      .BLKB 72
00200 DIAG_STORAGE_END:
      .BLKB 0
00200 FLAGS: .BLKB 4
00204 IMAGE_DUMP_STSFLG:
      .BLKB 4
00208 THIS_SYSID:
      .BLKB 6
0020E .BLKB 2
00210 CUR_TIME:
      .BLKB 8
00218 HOURLY_TIME:
      .BLKB 8
00220 HOURLY_PARAMS:
      .BLKB 20
00234 SYMBIONT_COUNT:
      .BLKB 4
00238 QUEUE_REFERENCE_COUNT:
      .BLKB -4

```

I 4  
15-Sep-1984 23:49:14  
14-Sep-1984 22:32:32

VAX-11 Bliss-32 V4.0-742  
[JOBCTL.SRC]ASYNCHRON.B32;3

Page 9  
(3)

```

0023C MBX_MESSAGE COUNT:
      .BLKB 4
00240 MBX: .BLKB 4
00244 MBX_END: .BLKB 4
00248 MEMORY_FREE_QUEUES:
      .BLKB 40
00270 NONAST_WORK_QUEUE:
      .BLKB 8
00278 BCB_FREE_LIST:
      .BLKB 4
0027C BCB_ACTIVE_LIST:
      .BLKB 4
00280 GQL_FREE_LIST:
      .BLKB 4
00284 GQL_ACTIVE_LIST:
      .BLKB 4
00288 OPEN_GETQUI_LIST:
      .BLKB 4
0028C PROCESS_DATA_LIST:
      .BLKB 4
00290 SYMBIONT_CONTROL:
      .BLKB 4
00294 SPARE_AREA:
      .BLKB 12
002A0 REMOTE_REQUEST_LKSB:
      .BLKB 8
002A8 QUEUE_FILE_LKSB:
      .BLKB 8
002B0 QUEUE_LOCK_LKSB:
      .BLKB 8
002B8 RSP: .BLKB 8
002C0 JBC_PRIORITY:
      .BLKB 4
002C4 JBC_PRIVILEGES:
      .BLKB 8
002CC JBC_QUOTAS:
      .BLKB 66
0030E .BLKB 2
00310 JBC_UIC: .BLKB 4
00314 QUEUE_FAB:
      .BLKB 80
00364 QUEUE_RAB:
      .BLKB 68
003A8 QUEUE_NAM:
      .BLKB 96
00408 QUEUE_XAB:
      .BLKB 88
00460 QUEUE_RSA:
      .BLKB 255
0055F .BLKB 1
00560 QUEUE_ALQ:
      .BLKB 4
00564 QUEUE_MBF:
      .BLKB 1
00565 .BLKB 3
00568 ACCOUNTING_FABS:
      .BLKB 8

```

J 4  
15-Sep-1984 23:49:14  
14-Sep-1984 22:32:32

VAX-11 BLISS-32 V4.0-742  
[JOBCTL.SRC]ASYNCHRON.B32;3

```

00570 ACCOUNTING_RABS:
      .BLKB 8
00578 ACCOUNT_FAB_A:
      .BLKB 80
005C8 ACCOUNT_RAB_A:
      .BLKB 68
0060C ACCOUNT_NAM_A:
      .BLKB 96
0066C ACCOUNT_RSA_A:
      .BLKB 255
0076B .BLKB 1
0076C ACCOUNT_FAB_B:
      .BLKB 80
007BC ACCOUNT_RAB_B:
      .BLKB 68
00800 ACCOUNT_NAM_B:
      .BLKB 96
00860 ACCOUNT_RSA_B:
      .BLKB 255
0095F .BLKB 1
00960 DIAG_FAB:
      .BLKB 80
009B0 DIAG_RAB:
      .BLKB 68
009F4 MBX_CHAN:
      .BLKB 4
009F8 MBX_IOSB:
      .BLKB 8
00A00 MBX_BUFFER:
      .BLKB 1024
00E00 VALUE_STORAGE_BASE:
      .BLKB 0
00E00 ITEM_PRESENT:
      .BLKB 32
00E20 VALUE_GETQUI_BASE:
      .BLKB 0
00E20 VALUE_ACCOUNTING_MESSAGE:
      .BLKB 6
00E26 VALUE_ACCOUNTING_TYPES:
      .BLKB 4
00E2A VALUE_AFTER_TIME:
      .BLKB 8
00E32 VALUE_ALIGNMENT_PAGES:
      .BLKB 1
00E33 VALUE_BASE_PRIORITY:
      .BLKB 1
00E34 VALUE_BATCH_INPUT:
      .BLKB 6
00E3A VALUE_BATCH_OUTPUT:
      .BLKB 10
00E44 VALUE_BUFFER_COUNT:
      .BLKB 1
00E45 VALUE_CHARACTERISTIC_NAME:
      .BLKB 6
00E4B VALUE_CHARACTERISTIC_NUMBER:
      .BLKB 1
00E4C VALUE_CHARACTERISTICS:

```

```

00E5C VALUE_CHECKPOINT_DATA:
      .BLKB 16
00E62 VALUE_CLI:
      .BLKB 8
00E68 VALUE_CPU_DEFAULT:
      .BLKB 6
00E6C VALUE_CPU_LIMIT:
      .BLKB 4
00E70 VALUE_DESTINATION_QUEUE:
      .BLKB 8
00E78 VALUE_DEVICE_NAME:
      .BLKB 6
00E7E VALUE_ENTRY_NUMBER:
      .BLKB 4
00E82 VALUE_ENTRY_NUMBER_OUTPUT:
      .BLKB 10
00E8C VALUE_EXTEND_QUANTITY:
      .BLKB 2
00E8E VALUE_FILE_COPIES:
      .BLKB 1
00E8F VALUE_FILE_IDENTIFICATION:
      .BLKB 36
00EB3 VALUE_FILE_SETUP_MODULES:
      .BLKB 8
00EB9 VALUE_FILE_SPECIFICATION:
      .BLKB 6
00EBF VALUE_FIRST_PAGE:
      .BLKB 4
00EC3 VALUE_FORM_DESCRIPTION:
      .BLKB 6
00EC9 VALUE_FORM_LENGTH:
      .BLKB 1
00ECA VALUE_FORM_MARGIN_BOTTOM:
      .BLKB 1
00ECB VALUE_FORM_MARGIN_LEFT:
      .BLKB 2
00ECD VALUE_FORM_MARGIN_RIGHT:
      .BLKB 2
00ECF VALUE_FORM_MARGIN_TOP:
      .BLKB 1
00ED0 VALUE_FORM_NAME:
      .BLKB 6
00ED6 VALUE_FORM_NUMBER:
      .BLKB 4
00EDA VALUE_FORM:
      .BLKB 8
00EE2 VALUE_FORM_SETUP_MODULES:
      .BLKB 8
00EE8 VALUE_FORM_STOCK:
      .BLKB 6
00EEE VALUE_FORM_WIDTH:
      .BLKB 2
00EF0 VALUE_GENERIC_TARGET:
      .BLKB 996
012D4 VALUE_JOB_COPIES:
      .BLKB 1

```

```

012D5 VALUE_JOB_LIMIT:
      .BLKB 1
012D6 VALUE_JOB_NAME:
      .BLKB 6
012DC VALUE_JOB_RESET_MODULES:
      .BLKB 6
012E2 VALUE_JOB_SIZE_MAXIMUM:
      .BLKB 4
012E6 VALUE_JOB_SIZE_MINIMUM:
      .BLKB 4
012EA VALUE_JOB_STATUS_OUTPUT:
      .BLKB 10
012F4 VALUE_LAST_PAGE:
      .BLKB 4
012F8 VALUE_LIBRARY_SPECIFICATION:
      .BLKB 6
012FE VALUE_LOG_QUEUE:
      .BLKB 8
01306 VALUE_LOG_SPECIFICATION:
      .BLKB 6
0130C VALUE_NOTE:
      .BLKB 6
01312 VALUE_OPERATOR_REQUEST:
      .BLKB 6
01318 VALUE_OWNER_UID:
      .BLKB 4
0131C VALUE_PAGE_SETUP_MODULES:
      .BLKB 8
01322 VALUE_PARAMETER_1:
      .BLKB 6
01328 VALUE_PARAMETER_2:
      .BLKB 6
0132E VALUE_PARAMETER_3:
      .BLKB 6
01334 VALUE_PARAMETER_4:
      .BLKB 6
0133A VALUE_PARAMETER_5:
      .BLKB 6
01340 VALUE_PARAMETER_6:
      .BLKB 6
01346 VALUE_PARAMETER_7:
      .BLKB 6
0134C VALUE_PARAMETER_8:
      .BLKB 6
01352 VALUE_PRIORITY:
      .BLKB 1
01353 VALUE_PROCESSOR:
      .BLKB 6
01359 VALUE_PROTECTION:
      .BLKB 4
0135D VALUE_QUEUE:
      .BLKB 6
01363 VALUE_QUEUE_FILE_SPECIFICATION:
      .BLKB 8
01369 VALUE_RELATIVE_PAGE:
      .BLKB 4
0136D VALUE_RESERVED_INPUT_1:

```

```

0136E VALUE_RESERVED_INPUT_2:
01370 VALUE_RESERVED_INPUT_3:
01374 VALUE_RESERVED_INPUT_4:
0137A VALUE_RESERVED_OUTPUT_1:
01384 VALUE_RESERVED_OUTPUT_2:
0138E VALUE_SEARCH_STRING:
01394 VALUE_SC$NODE_NAME:
0139A VALUE_WSDEFAULT:
0139C VALUE_W$EXTENT:
0139E VALUE_WSQUOTA:
013A0 VALUE_STORAGE_END:

```

```

JBC$_CLOSEOUT= 266328
JBC$_NOCMKRNL= 272388
JBC$_NUOPER= 272532
JBC$_NOSYSNAM= 272404
JBC$_OPENIN= 266392
JBC$_OPENOUT= 266400
JBC$_READERR= 266416
JBC$_WRITEERR= 266448

```

```

.EXTRN ABORT_EXECUTION
.EXTRN AFTER_AST, ALLOCATE_MEMORY
.EXTRN ALLOCATE_RECORD
.EXTRN BROADCAST_MESSAGE
.EXTRN COMPLETE_JOB, CREATE_SRB
.EXTRN DEALLOCATE_MEMORY
.EXTRN DEALLOCATE_RECORD
.EXTRN DELETE_FILES, FIND_PENDING_JOBS
.EXTRN FLUSH_RECORD, LOCK_QUEUE_FILE
.EXTRN PAUSE_EXECUTION
.EXTRN READ_RECORD, RELEASE_RECORD
.EXTRN RESET_EXECUTOR_QUEUE
.EXTRN RESUME_EXECUTION
.EXTRN REWRITE_RECORD, SCHEDULE_NONAST
.EXTRN SEND_SERVICE_RESPONSE_MESSAGE
.EXTRN START_EXECUTION
.EXTRN START_SYMBIONT_STREAM
.EXTRN STOP_SYMBIONT_STREAM
.EXTRN UNLOCK_QUEUE_FILE
.EXTRN UPDATE_GETQUI_DATA

```

```

.PSECT CODE, NOWRT, 2

```

OCFC 00000

```

.ENTRY CREATE_SRQ_RECORD, Save R2,R3,R4,R5,R6,R7,- ; 1194
R10,R11

```

Address	Hex	Label	Instruction	Comment	Page
0064	00BA		MOVAB	REWRITE_RECORD, R7	1233
0104	00BA		MOVAB	THIS_SYSID, R6	1234
	0104		CALLS	#0, ALLOCATE_RECORD	
			BLBS	STATUS, 1\$	
			RET		
			MOVB	#9, 4(SRQ)	1239
			MOVL	FUNC, 12(SRQ)	1240
			MOVL	THIS_SYSID, 20(SRQ)	1241
			MOVW	THIS_SYSID+4, 24(SRQ)	
			CASEL	FUNC, #1, #11	1242
			.WORD	3\$-2\$,-	
				4\$-2\$,-	
				7\$-2\$,-	
				8\$-2\$,-	
				9\$-2\$,-	
				10\$-2\$,-	
				11\$-2\$,-	
				13\$-2\$,-	
				15\$-2\$,-	
				20\$-2\$,-	
				16\$-2\$,-	
				17\$-2\$	
			BRB	6\$	
			BISB2	#1, 16(SRQ)	1260
			MOVL	SJH, R0	1261
			BISB2	#16, 17(R0)	
			BRB	5\$	1262
			MOVL	SJH, R0	1276
			BISB2	#2, 16(R0)	
			MOVL	SMQ, R0	1277
			MOVL	262(R0), 26(SRQ)	
			MOVW	266(R0), 30(SRQ)	
			MOVL	SMQ_N, 32(SRQ)	1278
			MOVL	SJH_N, 36(SRQ)	1279
			BRB	12\$	1244
			MOVL	SJH, R0	1289
			BISB2	#32, 17(R0)	
			BISB2	#2, 16(SRQ)	1290
			MOVL	THIS_SYSID, 26(SRQ)	1291
			MOVW	THIS_SYSID+4, 30(SRQ)	
			BRW	19\$	1292
			MOVL	SMQ, R0	1302
			BISB2	#1, 17(R0)	
			BICW2	#516, 16(R0)	1303
			BRB	14\$	1304
			MOVL	SMQ, R0	1315
			BISB2	#4, 17(R0)	
			BRB	14\$	1316
			MOVL	SMQ, R0	1327
			BISB2	#8, 16(R0)	
			BRB	14\$	1328
			MOVL	SMQ, R0	1344
			BISB2	#64, 16(R0)	
			MOVL	262(R0), 26(SRQ)	1345
			MOVW	266(R0), 30(SRQ)	
			MOVL	SMQ_N, 32(SRQ)	1346
			MOVQ	FLAGS, 36(SRQ)	1347

		2C	AB	18	AC	D0	000D9	MOVL	RELATIVE, 44(SRQ)	1349		
		30	AB	1C	AC	90	000DE	MOVB	SEARCH_LEN, 48(SRQ)	1350		
31	AB	20	BC	1C	AC	28	000E3	MOVC3	SEARCH_LEN, @SEARCH_ADDR, 49(SRQ)	1351		
				63	11	000EA	12\$:	BRB	20\$	1244		
			50	0C	AC	D0	000EC	13\$:	MOVL	SMQ, R0	1361	
		10	A0		20	88	000F0	BISB2	#32, 16(R0)			
				48	11	000F4	14\$:	BRB	18\$	1362		
		10	AB		01	88	000F6	15\$:	BISB2	#1, 16(SRQ)	1375	
			50	08	AC	D0	000FA	MOVL	SYSID, R0	1376		
		1A	AB		60	D0	000FE	MOVL	(R0), 26(SRQ)			
20	AB	1E	AB	04	A0	B0	00102	MOVW	4(R0), 30(SRQ)			
		0C	BC		0C	28	00107	MOVC3	#12, @USERNAME, 32(SRQ)	1377		
42	AB	40	AB	10	AC	B0	0010D	MOVW	LENGTH, 64(SRQ)	1378		
		14	BC	10	AC	28	00112	MOVC3	LENGTH, @ADDRESS, 66(SRQ)	1379		
					34	11	00119	BRB	20\$	1244		
		10	AB		01	88	0011B	16\$:	BISB2	#1, 16(SRQ)	1389	
			50	08	AC	D0	0011F	MOVL	SJH, R0	1390		
		1A	AB	016C	C0	D0	00123	MOVL	364(R0), 26(SRQ)			
		1E	AB	0170	C0	B0	00129	MOVW	368(R0), 30(SRQ)			
		20	AB	0C	AC	D0	0012F	MOVL	SQR_N, 32(SRQ)	1391		
					19	11	00134	BRB	20\$	1244		
		10	AB		02	88	00136	17\$:	BISB2	#2, 16(SRQ)	1401	
			50	0C	AC	D0	0013A	MOVL	SMQ, R0	1402		
		1A	AB	0106	C0	D0	0013E	18\$:	MOVL	262(R0), 26(SRQ)		
		1E	AB	010A	C0	B0	00144	MOVW	266(R0), 30(SRQ)			
		20	AB	08	AC	D0	0014A	19\$:	MOVL	SMQ_N, 32(SRQ)	1403	
			0A	10	AB	E8	0014F	20\$:	BLBS	16(SRQ), 21\$	1410	
				70	AB	9F	00153	PUSHAB	112(SRQ)	1412		
		00000000G	EF		01	FB	00156	CALLS	#1, CREATE_SRB			
		1A	AB		66	D1	0015D	21\$:	CMPL	THIS_SYSID, 26(SRQ)	1417	
					07	12	00161	BNEQ	22\$			
		1E	AB	04	A6	B1	00163	CMPL	THIS_SYSID+4, 30(SRQ)			
					0D	13	00168	BEQL	23\$			
08		10	AB		01	E0	0016A	22\$:	BBS	#1, 16(SRQ), 23\$	1418	
				1A	AB	9F	0016F	PUSHAB	26(SRQ)	1420		
		0000V	CF		01	FB	00172	CALLS	#1, ENTER_REMOTE_REQUEST			
					01	DD	00177	23\$:	PUSHL	#1	1425	
		00000000G	EF		01	FB	00179	CALLS	#1, READ_RECORD			
			6B	44	A0	D0	00180	MOVL	68(SQH), -(SRQ)	1426		
		44	A0		5A	D0	00184	MOVL	SRQ_N, 68(SQH)	1427		
					5A	DD	00188	PUSHL	SRQ_N	1428		
			67		01	FB	0018A	CALLS	#1, REWRITE_RECORD			
					01	DD	0018D	PUSHL	#1	1429		
			67		01	FB	0018F	CALLS	#1, REWRITE_RECORD			
					50	D4	00192	CLRL	R0	1435		
					04	00	00194	RET				

; Routine Size: 405 bytes, Routine Base: CODE + 0000

```

399 1436 1 ROUTINE PROCESS_REMOTE_SERVICES(SRQ;NEXT_ACTION): L_OUTPUT_1=
400 1437 1
401 1438 1 ++
402 1439 1
403 1440 1 FUNCTIONAL DESCRIPTION:
404 1441 1 This routine processes a remote service directed to this node.
405 1442 1
406 1443 1 INPUT PARAMETERS:
407 1444 1 SRQ - Pointer to SRQ.
408 1445 1
409 1446 1 IMPLICIT INPUTS:
410 1447 1 NONE
411 1448 1
412 1449 1 OUTPUT PARAMETERS:
413 1450 1 NEXT_ACTION - Code identifying the next action.
414 1451 1
415 1452 1 IMPLICIT OUTPUTS:
416 1453 1 NONE
417 1454 1
418 1455 1 ROUTINE VALUE:
419 1456 1 Completion status.
420 1457 1
421 1458 1 SIDE EFFECTS:
422 1459 1 NONE
423 1460 1
424 1461 1 --
425 1462 1
426 1463 2 BEGIN
427 1464 2 MAP
428 1465 2 SRQ: REF BBLOCK; ! Pointer to SRQ
429 1466 2 LOCAL
430 1467 2 STATUS; ! Status of the request
431 1468 2
432 1469 2
433 1470 2 STATUS = SS$NORMAL;
434 1471 2
435 1472 2
436 1473 2 CASE .SRQ[SRQ$L_FUNCTION_CODE] FROM SRQ$K_START_JOB TO SRQ$K_DELETE_FILES OF
437 1474 2 SET
438 1475 2
439 1476 2
440 1477 2 [INRANGE, OUTRANGE]:
441 1478 2 NEXT_ACTION = K_COMPLETE;
442 1479 2
443 1480 2
444 1481 2 [SRQ$K_START_JOB]:
445 1482 2 BEGIN
446 1483 2 LOCAL
447 1484 2 SMQ_N, ! Record number of SMQ
448 1485 2 SMQ: REF BBLOCK, ! Pointer to SMQ
449 1486 2 SJH_NP, ! Record number of predecessor of SJH
450 1487 2 SJH_P: REF BBLOCK, ! Predecessor of SJH
451 1488 2 SJH_N, ! Record number of SJH
452 1489 2 SJH: REF BBLOCK; ! Pointer to SJH
453 1490 2
454 1491 2
455 1492 2 SMQ = READ_RECORD(SMQ_N = SJH_NP = .SRQ[SRQ$L_P1]);

```

```

: 456      1493 3      SJH_N = .SMQ[SMQ$L_CURRENT_LIST];
: 457      1494 3      WHILE .SJH_N NEQ 0 DO
: 458      1495 4      BEGIN
: 459      1496 4      SJH = READ_RECORD(.SJH_N);
: 460      1497 4      IF .SJH_N EQL .SRQ[SRQ$L_P2]
: 461      1498 4      THEN
: 462      1499 5      BEGIN
: 463      1500 5      SJH[SJH$V_STARTING] = FALSE;
: 464      1501 5      STATUS = START_EXECUTION(
: 465      1502 5      .SMQ_N, .SMQ,
: 466      1503 5      .SJH_N, .SJH);
: 467      1504 5      IF NOT .STATUS
: 468      1505 5      THEN
: 469      1506 6      BEGIN
: 470      1507 6      UPDATE GETQUI DATA(.SJH_N, .SJH);
: 471      1508 6      SMQ[SMQ$B_CURRENT_JOB_COUNT] = .SMQ[SMQ$B_CURRENT_JOB_COUNT] - 1;
: 472      1509 6      IF .SJH_NP EQL .SMQ_N
: 473      1510 6      THEN
: 474      1511 7      BEGIN
: 475      1512 7      SMQ[SMQ$L_CURRENT_LIST] = .SJH[SYM$L_LINK];
: 476      1513 7      IF .SJH[SYM$L_LINK] EQL 0 THEN SMQ[SMQ$L_CURRENT_LIST_END] = 0;
: 477      1514 7      END
: 478      1515 6      ELSE
: 479      1516 7      BEGIN
: 480      1517 7      SJH_P[SYM$L_LINK] = .SJH[SYM$L_LINK];
: 481      1518 7      IF .SJH[SYM$L_LINK] EQL 0 THEN SMQ[SMQ$L_CURRENT_LIST_END] = .SJH_NP;
: 482      1519 7      REWRITE_RECORD(.SJH_NP);
: 483      1520 6      END;
: 484      1521 6      SJH[SJH$L_CONDITION_1] = .STATUS;
: 485      1522 6      COMPLETE_JOB(.SJH_N, .SJH, .SMQ, 0);
: 486      1523 6      FIND_PENDING_JOBS(.SMQ_N, .SMQ);
: 487      1524 6      END
: 488      1525 5      ELSE
: 489      1526 5      REWRITE_RECORD(.SJH_N);
: 490      1527 5      REWRITE_RECORD(.SMQ_N);
: 491      1528 5      EXITLOOP;
: 492      1529 5      END;
: 493      1530 4      IF .SJH_NP NEQ .SMQ_N THEN RELEASE_RECORD(.SJH_NP);
: 494      1531 4      SJH_NP = .SJH_N;
: 495      1532 4      SJH_P = .SJH;
: 496      1533 4      SJH_N = .SJH[SYM$L_LINK];
: 497      1534 4      END;
: 498      1535 3      NEXT_ACTION = K_DEALLOCATE;
: 499      1536 3      END;
: 500      1537 2
: 501      1538 2
: 502      1539 2
: 503      1540 2      [SRQ$K_ABORT_JOB]:
: 504      1541 3      BEGIN
: 505      1542 3      LOCAL
: 506      1543 3      SMQ_N,
: 507      1544 3      SMQ:      REF BBLOCK,      ! Record number of SMQ
: 508      1545 3      SJH_N,      ! Pointer to SMQ
: 509      1546 3      SJH_NP,      ! Record number of SJH
: 510      1547 3      SJH:      REF BBLOCK;      ! Successor of SJH
: 511      1548 3
: 512      1549 3

```

```

: 513      1550      3      SMQ = READ_RECORD(SMQ_N = .SRQ[SRQ$L_P1]);
: 514      1551      3      SJH_N = .SMQ[SMQ$L_CURRENT_LIST];
: 515      1552      3      WHILE .SJH_N NEQ 0 DO
: 516      1553      4      BEGIN
: 517      1554      4      SJH = READ_RECORD(.SJH_N);
: 518      1555      4      IF .SJH_N EQL .SRQ[SRQ$L_P2]
: 519      1556      4      THEN
: 520      1557      5      BEGIN
: 521      1558      5      SJH[SJH$V_ABORTING] = FALSE;
: 522      1559      5      STATUS = ABORT_EXECUTION(
: 523      1560      5      .SMQ_N, .SMQ,
: 524      1561      5      .SJH_N, .SJH);
: 525      1562      5      REWRITE_RECORD(.SJH_N);
: 526      1563      5      EXITLOOP;
: 527      1564      4      END;
: 528      1565      4      SJH_NS = .SJH[SYM$L_LINK];
: 529      1566      4      RELEASE_RECORD(.SJH_N);
: 530      1567      4      SJH_N = .SJH_NS;
: 531      1568      3      END;
: 532      1569      3      NEXT_ACTION = K_COMPLETE;
: 533      1570      2      END;
: 534      1571      2
: 535      1572      2      [SRQ$K_START_QUEUE]:
: 536      1573      2      BEGIN
: 537      1574      3      LOCAL
: 538      1575      3      SMQ_N,
: 539      1576      3      SMQ:
: 540      1577      3      REF BBLOCK;
: 541      1578      3      ! Record number of SMQ
: 542      1579      3      ! Pointer to SMQ
: 543      1580      3
: 544      1581      3      SMQ = READ_RECORD(SMQ_N = .SRQ[SRQ$L_P1]);
: 545      1582      3      STATUS = START_SYMBIONT_STREAM(.SMQ_N, .SMQ);
: 546      1583      3      IF NOT .STATUS
: 547      1584      4      THEN
: 548      1585      4      BEGIN
: 549      1586      4      SMQ[SMQ$V_STARTING] = FALSE;
: 550      1587      4      SMQ[SMQ$V_STOPPED] = TRUE;
: 551      1588      4      NEXT_ACTION = K_COMPLETE;
: 552      1589      4      END
: 553      1590      3      ELSE
: 554      1591      4      BEGIN
: 555      1592      4      SRQ[SRQ$L_FUNCTION_CODE] = SRQ$K_START_SYMBIONT;
: 556      1593      4      SRQ[SRQ$V_STALLED] = TRUE;
: 557      1594      3      NEXT_ACTION = K_REWRITE;
: 558      1595      3      END;
: 559      1596      2      REWRITE_RECORD(.SMQ_N);
: 560      1597      2      END;
: 561      1598      2
: 562      1599      2      [SRQ$K_STOP_QUEUE]:
: 563      1600      2      BEGIN
: 564      1601      3      LOCAL
: 565      1602      3      SMQ_N,
: 566      1603      3      SMQ:
: 567      1604      3      REF BBLOCK;
: 568      1605      3      ! Record number of SMQ
: 569      1606      3      ! Pointer to SMQ
:
:      SMQ = READ_RECORD(SMQ_N = .SRQ[SRQ$L_P1]);
```

```

570      SMQ[SMQ$V STOPPING] = FALSE;
571      STOP SYMBIONT STREAM(.SMQ_N, .SMQ);
572      REWRITE RECORD(.SMQ_N);
573      NEXT_ACTION = K_COMPLETE;
574      END;
575
576      [SRQ$K PAUSE_QUEUE]:
577      BEGIN
578      LOCAL
579          SMQ_N,
580          SMQ:
581          REF BBLOCK;
582          ! Record number of SMQ
583          ! Pointer to SMQ
584
585      SMQ = READ RECORD(SMQ_N = .SRQ[SRQ$L_P1]);
586      SMQ[SMQ$V PAUSING] = FALSE;
587      STATUS = PAUSE_EXECUTION(.SMQ_N, .SMQ);
588      IF NOT .STATUS THEN FIND_PENDING_JOBS(.SMQ_N, .SMQ);
589      REWRITE RECORD(.SMQ_N);
590      NEXT_ACTION = K_COMPLETE;
591      END;
592
593      [SRQ$K RESUME_QUEUE]:
594      BEGIN
595      LOCAL
596          SMQ_N,
597          SMQ:
598          REF BBLOCK;
599          ! Record number of SMQ
600          ! Pointer to SMQ
601
602      SMQ = READ RECORD(SMQ_N = .SRQ[SRQ$L_P1]);
603      SMQ[SMQ$V RESUMING] = FALSE;
604      STATUS = RESUME_EXECUTION(
605          .SMQ_N, .SMQ,
606          .SRQ[SRQ$L_P2], .SRQ[SRQ$L_P3], .SRQ[SRQ$L_P4],
607          CHRCHAR(SRQ[SRQ$T_P5], SRQ[SRQ$T_P5]+1);
608      FIND_PENDING_JOBS(.SMQ_N, .SMQ);
609      REWRITE RECORD(.SMQ_N);
610      NEXT_ACTION = K_COMPLETE;
611      END;
612
613      [SRQ$K RESET_QUEUE]:
614      BEGIN
615      LOCAL
616          SMQ_N,
617          SMQ:
618          REF BBLOCK;
619          ! Record number of SMQ
620          ! Pointer to SMQ
621
622      SMQ[SMQ$V RESETING] = FALSE;
623      SMQ = READ RECORD(SMQ_N = .SRQ[SRQ$L_P1]);
624      RESET_EXECUTOR_QUEUE(.SMQ_N, .SMQ);
625      REWRITE RECORD(.SMQ_N);
626      NEXT_ACTION = K_COMPLETE;
627      END;
```

```

: 627      1664 2      [SRQ$K_BROADCAST_MESSAGE]:
: 628      1665      BEGIN
: 629      1666      BROADCAST_MESSAGE(
: 630      1667          THIS_SYSID,
: 631      1668          SRQ[SRQ$T_BRDCST_USERNAME],
: 632      1669          .SRQ[SRQ$W_BRDCST_LENGTH],
: 633      1670          SRQ[SRQ$T_BRDCST_TEXT]);
: 634      1671      NEXT_ACTION = K_DEALLOCATE;
: 635      1672      END;
: 636      1673
: 637      1674
: 638      1675      [SRQ$K_RESPONSE]:
: 639      1676      BEGIN
: 640      1677      SEND_SERVICE_RESPONSE_MESSAGE(
: 641      1678          SRQ[SRQ$T_SRB],
: 642      1679          .SRQ[SRQ$C_P1]);
: 643      1680      NEXT_ACTION = R_DEALLOCATE;
: 644      1681      END;
: 645      1682
: 646      1683
: 647      1684      [SRQ$K_DELETE_FILES]:
: 648      1685      BEGIN
: 649      1686      DELETE_FILES(.SRQ[SRQ$L_P1]);
: 650      1687      NEXT_ACTION = K_DEALLOCATE;
: 651      1688      END;
: 652      1689
: 653      1690
: 654      1691      TES;
: 655      1692
: 656      1693
: 657      1694      .STATUS
: 658      1695      1 END;
: 659      1696
: 660      1697
: 661      1698
: 662      1699
: 663      1700
: 664      1701
: 665      1702
: 666      1703
: 667      1704
: 668      1705
: 669      1706
: 670      1707
: 671      1708
: 672      1709
: 673      1710
: 674      1711
: 675      1712
: 676      1713
: 677      1714
: 678      1715
: 679      1716
: 680      1717
: 681      1718
: 682      1719
: 683      1720
: 684      1721
: 685      1722
: 686      1723
: 687      1724
: 688      1725
: 689      1726
: 690      1727
: 691      1728
: 692      1729
: 693      1730
: 694      1731
: 695      1732
: 696      1733
: 697      1734
: 698      1735
: 699      1736
: 700      1737
: 701      1738
: 702      1739
: 703      1740
: 704      1741
: 705      1742
: 706      1743
: 707      1744
: 708      1745
: 709      1746
: 710      1747
: 711      1748
: 712      1749
: 713      1750
: 714      1751
: 715      1752
: 716      1753
: 717      1754
: 718      1755
: 719      1756
: 720      1757
: 721      1758
: 722      1759
: 723      1760
: 724      1761
: 725      1762
: 726      1763
: 727      1764
: 728      1765
: 729      1766
: 730      1767
: 731      1768
: 732      1769
: 733      1770
: 734      1771
: 735      1772
: 736      1773
: 737      1774
: 738      1775
: 739      1776
: 740      1777
: 741      1778
: 742      1779
: 743      1780
: 744      1781
: 745      1782
: 746      1783
: 747      1784
: 748      1785
: 749      1786
: 750      1787
: 751      1788
: 752      1789
: 753      1790
: 754      1791
: 755      1792
: 756      1793
: 757      1794
: 758      1795
: 759      1796
: 760      1797
: 761      1798
: 762      1799
: 763      1800
: 764      1801
: 765      1802
: 766      1803
: 767      1804
: 768      1805
: 769      1806
: 770      1807
: 771      1808
: 772      1809
: 773      1810
: 774      1811
: 775      1812
: 776      1813
: 777      1814
: 778      1815
: 779      1816
: 780      1817
: 781      1818
: 782      1819
: 783      1820
: 784      1821
: 785      1822
: 786      1823
: 787      1824
: 788      1825
: 789      1826
: 790      1827
: 791      1828
: 792      1829
: 793      1830
: 794      1831
: 795      1832
: 796      1833
: 797      1834
: 798      1835
: 799      1836
: 800      1837
: 801      1838
: 802      1839
: 803      1840
: 804      1841
: 805      1842
: 806      1843
: 807      1844
: 808      1845
: 809      1846
: 810      1847
: 811      1848
: 812      1849
: 813      1850
: 814      1851
: 815      1852
: 816      1853
: 817      1854
: 818      1855
: 819      1856
: 820      1857
: 821      1858
: 822      1859
: 823      1860
: 824      1861
: 825      1862
: 826      1863
: 827      1864
: 828      1865
: 829      1866
: 830      1867
: 831      1868
: 832      1869
: 833      1870
: 834      1871
: 835      1872
: 836      1873
: 837      1874
: 838      1875
: 839      1876
: 840      1877
: 841      1878
: 842      1879
: 843      1880
: 844      1881
: 845      1882
: 846      1883
: 847      1884
: 848      1885
: 849      1886
: 850      1887
: 851      1888
: 852      1889
: 853      1890
: 854      1891
: 855      1892
: 856      1893
: 857      1894
: 858      1895
: 859      1896
: 860      1897
: 861      1898
: 862      1899
: 863      1900
: 864      1901
: 865      1902
: 866      1903
: 867      1904
: 868      1905
: 869      1906
: 870      1907
: 871      1908
: 872      1909
: 873      1910
: 874      1911
: 875      1912
: 876      1913
: 877      1914
: 878      1915
: 879      1916
: 880      1917
: 881      1918
: 882      1919
: 883      1920
: 884      1921
: 885      1922
: 886      1923
: 887      1924
: 888      1925
: 889      1926
: 890      1927
: 891      1928
: 892      1929
: 893      1930
: 894      1931
: 895      1932
: 896      1933
: 897      1934
: 898      1935
: 899      1936
: 900      1937
: 901      1938
: 902      1939
: 903      1940
: 904      1941
: 905      1942
: 906      1943
: 907      1944
: 908      1945
: 909      1946
: 910      1947
: 911      1948
: 912      1949
: 913      1950
: 914      1951
: 915      1952
: 916      1953
: 917      1954
: 918      1955
: 919      1956
: 920      1957
: 921      1958
: 922      1959
: 923      1960
: 924      1961
: 925      1962
: 926      1963
: 927      1964
: 928      1965
: 929      1966
: 930      1967
: 931      1968
: 932      1969
: 933      1970
: 934      1971
: 935      1972
: 936      1973
: 937      1974
: 938      1975
: 939      1976
: 940      1977
: 941      1978
: 942      1979
: 943      1980
: 944      1981
: 945      1982
: 946      1983
: 947      1984
: 948      1985
: 949      1986
: 950      1987
: 951      1988
: 952      1989
: 953      1990
: 954      1991
: 955      1992
: 956      1993
: 957      1994
: 958      1995
: 959      1996
: 960      1997
: 961      1998
: 962      1999
: 963      2000
: 964      2001
: 965      2002
: 966      2003
: 967      2004
: 968      2005
: 969      2006
: 970      2007
: 971      2008
: 972      2009
: 973      2010
: 974      2011
: 975      2012
: 976      2013
: 977      2014
: 978      2015
: 979      2016
: 980      2017
: 981      2018
: 982      2019
: 983      2020
: 984      2021
: 985      2022
: 986      2023
: 987      2024
: 988      2025
: 989      2026
: 990      2027
: 991      2028
: 992      2029
: 993      2030
: 994      2031
: 995      2032
: 996      2033
: 997      2034
: 998      2035
: 999      2036
: 1000     2037
: 1001     2038
: 1002     2039
: 1003     2040
: 1004     2041
: 1005     2042
: 1006     2043
: 1007     2044
: 1008     2045
: 1009     2046
: 1010     2047
: 1011     2048
: 1012     2049
: 1013     2050
: 1014     2051
: 1015     2052
: 1016     2053
: 1017     2054
: 1018     2055
: 1019     2056
: 1020     2057
: 1021     2058
: 1022     2059
: 1023     2060
: 1024     2061
: 1025     2062
: 1026     2063
: 1027     2064
: 1028     2065
: 1029     2066
: 1030     2067
: 1031     2068
: 1032     2069
: 1033     2070
: 1034     2071
: 1035     2072
: 1036     2073
: 1037     2074
: 1038     2075
: 1039     2076
: 1040     2077
: 1041     2078
: 1042     2079
: 1043     2080
: 1044     2081
: 1045     2082
: 1046     2083
: 1047     2084
: 1048     2085
: 1049     2086
: 1050     2087
: 1051     2088
: 1052     2089
: 1053     2090
: 1054     2091
: 1055     2092
: 1056     2093
: 1057     2094
: 1058     2095
: 1059     2096
: 1060     2097
: 1061     2098
: 1062     2099
: 1063     2100
: 1064     2101
: 1065     2102
: 1066     2103
: 1067     2104
: 1068     2105
: 1069     2106
: 1070     2107
: 1071     2108
: 1072     2109
: 1073     2110
: 1074     2111
: 1075     2112
: 1076     2113
: 1077     2114
: 1078     2115
: 1079     2116
: 1080     2117
: 1081     2118
: 1082     2119
: 1083     2120
: 1084     2121
: 1085     2122
: 1086     2123
: 1087     2124
: 1088     2125
: 1089     2126
: 1090     2127
: 1091     2128
: 1092     2129
: 1093     2130
: 1094     2131
: 1095     2132
: 1096     2133
: 1097     2134
: 1098     2135
: 1099     2136
: 1100     2137
: 1101     2138
: 1102     2139
: 1103     2140
: 1104     2141
: 1105     2142
: 1106     2143
: 1107     2144
: 1108     2145
: 1109     2146
: 1110     2147
: 1111     2148
: 1112     2149
: 1113     2150
: 1114     2151
: 1115     2152
: 1116     2153
: 1117     2154
: 1118     2155
: 1119     2156
: 1120     2157
: 1121     2158
: 1122     2159
: 1123     2160
: 1124     2161
: 1125     2162
: 1126     2163
: 1127     2164
: 1128     2165
: 1129     2166
: 1130     2167
: 1131     2168
: 1132     2169
: 1133     2170
: 1134     2171
: 1135     2172
: 1136     2173
: 1137     2174
: 1138     2175
: 1139     2176
: 1140     2177
: 1141     2178
: 1142     2179
: 1143     2180
: 1144     2181
: 1145     2182
: 1146     2183
: 1147     2184
: 1148     2185
: 1149     2186
: 1150     2187
: 1151     2188
: 1152     2189
: 1153     2190
: 1154     2191
: 1155     2192
: 1156     2193
: 1157     2194
: 1158     2195
: 1159     2196
: 1160     2197
: 1161     2198
: 1162     2199
: 1163     2200
: 1164     2201
: 1165     2202
: 1166     2203
: 1167     2204
: 1168     2205
: 1169     2206
: 1170     2207
: 1171     2208
: 1172     2209
: 1173     2210
: 1174     2211
: 1175     2212
: 1176     2213
: 1177     2214
: 1178     2215
: 1179     2216
: 1180     2217
: 1181     2218
: 1182     2219
: 1183     2220
: 1184     2221
: 1185     2222
: 1186     2223
: 1187     2224
: 1188     2225
: 1189     2226
: 1190     2227
: 1191     2228
: 1192     2229
: 1193     2230
: 1194     2231
: 1195     2232
: 1196     2233
: 1197     2234
: 1198     2235
: 1199     2236
: 1200     2237
: 1201     2238
: 1202     2239
: 1203     2240
: 1204     2241
: 1205     2242
: 1206     2243
: 1207     2244
: 1208     2245
: 1209     2246
: 1210     2247
: 1211     2248
: 1212     2249
: 1213     2250
: 1214     2251
: 1215     2252
: 1216     2253
: 1217     2254
: 1218     2255
: 1219     2256
: 1220     2257
: 1221     2258
: 1222     2259
: 1223     2260
: 1224     2261
: 1225     2262
: 1226     2263
: 1227     2264
: 1228     2265
: 1229     2266
: 1230     2267
: 1231     2268
: 1232     2269
: 1233     2270
: 1234     2271
: 1235     2272
: 1236     2273
: 1237     2274
: 1238     2275
: 1239     2276
: 1240     2277
: 1241     2278
: 1242     2279
: 1243     2280
: 1244     2281
: 1245     2282
: 1246     2283
: 1247     2284
: 1248     2285
: 1249     2286
: 1250     2287
: 1251     2288
: 1252     2289
: 1253     2290
: 1254     2291
: 1255     2292
: 1256     2293
: 1257     2294
: 1258     2295
: 1259     2296
: 1260     2297
: 1261     2298
: 1262     2299
: 1263     2300
: 1264     2301
: 1265     2302
: 1266     2303
: 1267     2304
: 1268     2305
: 1269     2306
: 1270     2307
: 1271     2308
: 1272     2309
: 1273     2310
: 1274     2311
: 1275     2312
: 1276     2313
: 1277     2314
: 1278     2315
: 1279     2316
: 1280     2317
: 1281     2318
: 1282     2319
: 1283     2320
: 1284     2321
: 1285     2322
: 1286     2323
: 1287     2324
: 1288     2325
: 1289     2326
: 1290     2327
: 1291     2328
: 1292     2329
: 1293     2330
: 1294     2331
: 1295     2332
: 1296     2333
: 1297     2334
: 1298     2335
: 1299     2336
: 1300     2337
: 1301     2338
: 1302     2339
: 1303     2340
: 1304     2341
: 1305     2342
: 1306     2343
: 1307     2344
: 1308     2345
: 1309     2346
: 1310     2347
: 1311     2348
: 1312     2349
: 1313     2350
: 1314     2351
: 1315     2352
: 1316     2353
: 1317     2354
: 1318     2355
: 1319     2356
: 1320     2357
: 1321     2358
: 1322     2359
: 1323     2360
: 1324     2361
: 1325     2362
: 1326     2363
: 1327     2364
: 1328     2365
: 1329     2366
: 1330     2367
: 1331     2368
: 1332     2369
: 1333     2370
: 1334     2371
: 1335     2372
: 1336     2373
: 1337     2374
: 1338     2375
: 1339     2376
: 1340     2377
: 1341     2378
: 1342     2379
: 1343     2380
: 1344     2381
: 1345     2382
: 1346     2383
: 1347     2384
: 1348     2385
: 1349     2386
: 1350     2387
: 1351     2388
: 1352     2389
: 1353     2390
: 1354     2391
: 1355     2392
: 1356     2393
: 1357     2394
: 1358     2395
: 1359     2396
: 1360     2397
: 1361     2398
: 1362     2399
: 1363     2400
: 1364     2401
: 1365     2402
: 1366     2403
: 1367     2404
: 1368     2405
: 1369     2406
: 1370     2407
: 1371     2408
: 1372     2409
: 1373     2410
: 1374     2411
: 1375     2412
: 1376     2413
: 1377     2414
: 1378     2415
: 1379     2416
: 1380     2417
: 1381     2418
: 1382     2419
: 1383     2420
: 1384     2421
: 1385     2422
: 1386     2423
: 1387     2424
: 1388     2425
: 1389     2426
: 1390     2427
: 1391     2428
: 1392     2429
: 1393     2430
: 1394     2431
: 1395     2432
: 1396     2433
: 1397     2434
: 1398     2435
: 1399     2436
: 1400     2437
: 1401     2438
: 1402     2439
: 1403     2440
: 1404     2441
: 1405     2442
: 1406     2443
: 1407     2444
: 1408     2445
: 1409     2446
: 1410     2447
: 1411     2448
: 1412     2449
: 1413     2450
: 1414     2451
: 1415     2452
: 1416     2453
: 1417     2454
: 1418     2455
: 1419     2456
: 1420     2457
: 1421     2458
: 1422     2459
: 1423     2460
: 1424     2461
: 1425     2462
: 1426     2463
: 1427     2464
: 1428     2465
: 1429     2466
: 1430     2467
: 1431     2468
: 1432     2469
: 1433     2470
: 1434     2471
: 1435     2472
: 1436     2473
: 1437     2474
: 1438     2475
: 1439     2476
: 1440     2477
: 1441     2478
: 1442     2479
: 1443     2480
: 1444     2481
: 1445     2482
: 1446     2483
: 1447     2484
: 1448     2485
: 1449     2486
: 1450     2487
: 1451     2488
: 1452     2489
: 1453     2490
: 1454     2491
: 1455     2492
: 1456     2493
: 1457     2494
: 1458     2495
: 1459     2496
: 1460     2497
: 1461     2498
: 1462     2499
: 1463     2500
: 1464     2501
: 1465     2502
: 1466     2503
: 1467     2504
: 1468     2505
: 1469     2506
: 1470     2507
: 1471     2508
: 1472     2509
: 1473     2510
: 1474     2511
: 1475     2512
: 1476     2513
: 1477     2514
: 1478     2515
: 1479     2516
: 1480     2517
: 1481     2518
: 1482     2519
: 1483     2520
: 1484     2521
: 1485     2522
: 1486     2523
: 1487     2524
: 1488     2525
: 1489     2526
: 1490     2527
: 1491     2528
: 1492     2529
: 1493     2530
: 1494     2531
: 1495     2532
: 1496     2533
: 1497     2534
: 1498     2535
: 1499     2536
: 1500     2537
: 1501     2538
: 1502     2539
: 1503     2540
: 1504     2541
: 1505     2542
: 1506     2543
: 1507     2544
: 1508     2545
: 1509     2546
: 1510     2547
: 1511     2548
: 1512     2549
: 1513     2550
: 1514     2551
: 1515     2552
: 1516     2553
: 1517     2554
: 1518     2555
: 1519     2556
: 1520     2557
: 1521     2558
: 1522     2559
: 1523     2560
: 1524     2561
: 1525     2562
: 1526     2563
: 1527     2564
: 1528     2565
: 1529     2566
: 1530     2567
: 1531     2568
: 1532     2569
: 1533     2570
: 1534     2571
: 1535     2572
: 1536     2573
: 1537     2574
: 1538     2575
: 1539     2576
: 1540     2577
: 1541     2578
: 1542     2579
: 1543     2580
: 1544     2581
: 1545     2582
: 1546     2583
: 1547     2584
: 1548     2585
: 1549     2586
: 1550     2587
: 1551     2588
: 1552     2589
: 1553     2590
: 1554     2591
: 1555     2592
: 1556     2593
: 1557     2594
: 1558     2595
: 1559     2596
: 1560     2597
: 1561     2598
: 1562     2599
: 1563     2600
: 1564     2601
: 1565     2602
: 1566     2603
: 1567     2604
: 1568     2605
: 1569     2606
: 1570     2607
:
```

			01E6	31	0002B		BRW	31\$-1\$		
	56	20	A2	D0	0002E	2\$:	MOV L	28\$		1478
	57		56	D0	00032		MOV L	32(R2), SJH_NP		1492
			56	DD	00035		MOV L	SJH_NP, SMQ_N		
	6A		01	FB	00037		PUSHL	SJH_NP		
	53		50	D0	0003A		CALLS	#1, READ_RECORD		
	55	48	A3	D0	0003D		MOV L	R0, SMQ		1493
			03	12	00041	3\$:	MOV L	72(SMQ), SJH_N		1494
			0204	31	00043		BNEQ	4\$		
			55	DD	00046	4\$:	BRW	32\$		1496
	6A		01	FB	00048		PUSHL	SJH_N		
	54		50	D0	0004B		CALLS	#1, READ_RECORD		
24	A2		55	D1	0004E		MOV L	R0, SJH		
			7E	12	00052		CMP L	SJH_N, 36(R2)		1497
11	A4		10	8A	00054		BNEQ	10\$		
			54	DD	00058		BICB2	#16, 17(SJH)		1500
			28	BB	0005A		PUSHL	SJH		1503
			57	DD	0005C		PUSHR	#^M<R3,R5>		1502
00000000G	EF		04	FB	0005E		PUSHL	SMQ_N		
	58		50	D0	00065		CALLS	#4, START_EXECUTION		
	52		58	E8	00068		MOV L	R0, STATUS		
			54	DD	0006B		BLBS	STATUS, 8\$		1504
			55	DD	0006D		PUSHL	SJH		1507
00000000G	EF		02	FB	0006F		PUSHL	SJH_N		
	57	0115	C3	97	00076		CALLS	#2, UPDATE_GETQUI_DATA		
			56	D1	0007A		DEC B	277(SMQ)		1508
			0B	12	0007D		CMP L	SJH_NP, SMQ_N		1509
	48	A3	64	D0	0007F		BNEQ	5\$		
			17	12	00083		MOV L	(SJH), 72(SMQ)		1512
		4C	A3	D4	00085		BNEQ	7\$		1513
			12	11	00088		CLRL	76(SMQ)		
	69		64	D0	0008A	5\$:	BRB	7\$		1509
			04	12	0008D		MOV L	(SJH), (SJH_P)		1517
	4C	A3	56	D0	0008F		BNEQ	6\$		1518
			56	DD	00093	6\$:	MOV L	SJH_NP, 76(SMQ)		
00000000G	EF		01	FB	00095		PUSHL	SJH_NP		1519
00DC	C4		58	D0	0009C	7\$:	CALLS	#1, REWRITE_RECORD		
			7E	D4	000A1		MOV L	STATUS, 220(SJH)		1521
			53	DD	000A3		CLRL	-(SP)		1522
			54	DD	000A5		PUSHL	SMQ		
			55	DD	000A7		PUSHL	SJH		
00000000G	EF		04	FB	000A9		PUSHL	SJH_N		
			53	DD	000B0		CALLS	#4, COMPLETE_JOB		
			57	DD	000B2		PUSHL	SMQ		1523
00000000G	EF		02	FB	000B4		PUSHL	SMQ_N		
			09	11	000BB		CALLS	#2, FIND_PENDING_JOBS		
			55	DD	000BD	8\$:	BRB	9\$		1504
00000000G	EF		01	FB	000BF		PUSHL	SJH_N		1526
			57	DD	000C6	9\$:	CALLS	#1, REWRITE_RECORD		
00000000G	EF		01	FB	000C8		PUSHL	SMQ_N		1528
			56	DD	000D7		CALLS	#1, REWRITE_RECORD		
	57	0178	31	000CF			BRW	32\$		1499
			56	D1	000D2	10\$:	CMP L	SJH_NP, SMQ_N		1531
			09	13	000D5		BEQ L	11\$		
			56	DD	000D9		PUSHL	SJH_NP		
00000000G	EF		01	FB	000D9		CALLS	#1, RELEASE_RECORD		
	56		55	D0	000E0	11\$:	MOV L	SJH_N, SJH_NP		1532

	59		54	DO	000E3	MOVL	SJH, SJH_P	: 1533
	55		64	DO	000E6	MOVL	(SJH), SJH_N	: 1534
			FF55	31	000E9	BRW	3\$	: 1494
	57	20	A2	DO	000EC	12\$:	MOVL 32(R2), SMQ_N	: 1550
			57	DD	000F0	PUSHL	SMQ_N	
	6A		01	FB	000F2	CALLS	#1, READ_RECORD	
	55		50	DO	000F5	MOVL	R0, SMQ	
	54	4R	A5	DO	000F8	MOVL	72(SMQ), SJH_N	: 1551
			03	12	000FC	13\$:	BNEQ 14\$	: 1552
			0113	31	000FE	BRW	28\$	
			54	DD	00101	14\$:	PUSHL SJH_N	: 1554
	6A		01	FB	00103	CALLS	#1, READ_RECORD	
	53		50	DO	00106	MOVL	R0, SJH	
24	A2		54	D1	00109	CMPL	SJH_N, 36(R2)	: 1555
			1B	12	0010D	BNEQ	16\$	
10	A3		02	8A	0010F	BICB2	#2, 16(SJH)	: 1558
			53	DD	00113	PUSHL	SJH	: 1561
			54	DD	00115	PUSHL	SJH_N	
			55	DD	00117	PUSHL	SMQ	: 1560
			57	DD	00119	PUSHL	SMQ_N	
00000000G	EF		04	FB	0011B	CALLS	#4, ABORT_EXECUTION	
	58		50	DO	00122	MOVL	R0, STATUS	
			54	DD	00125	15\$:	PUSHL SJH_N	: 1562
			00E3	31	00127	BRW	27\$	
	56		63	DO	0012A	16\$:	MOVL (SJH), SJH_NS	: 1565
			54	DD	0012D	PUSHL	SJH_N	: 1566
00000000G	EF		01	FB	0012F	CALLS	#1, RELEASE_RECORD	
	54		56	DO	00136	MOVL	SJH_NS, SJH_N	: 1567
			C1	11	00139	BRB	13\$	: 1552
	54	20	A2	DO	0013B	17\$:	MOVL 32(R2), SMQ_N	: 1580
			54	DD	0013F	PUSHL	SMQ_N	
	6A		01	FB	00141	CALLS	#1, READ_RECORD	
	53		50	DO	00144	MOVL	R0, SMQ	
			53	DD	00147	PUSHL	SMQ	: 1581
			54	DD	00149	PUSHL	SMQ_N	
00000000G	EF		02	FB	0014B	CALLS	#2, START_SYMBIONT_STREAM	
	58		50	DO	00152	MOVL	R0, STATUS	
	0C		58	E8	00155	BLBS	STATUS, 18\$	: 1582
11	A3		01	8A	00158	BICB2	#1, 17(SMQ)	: 1585
11	A3		02	88	0015C	BISB2	#2, 17(SMQ)	: 1586
			5B	D4	00160	CLRL	NEXT_ACTION	: 1587
			0B	11	00162	BRB	19\$	: 1582
	0C	A2	0C	DO	00164	18\$:	MOVL #12, 12(R2)	: 1591
	10	A2	02	88	00168	BISB2	#2, 16(R2)	: 1592
		5B	03	DO	0016C	MOVL	#3, NEXT_ACTION	: 1593
			54	DD	0016F	19\$:	PUSHL SMQ_N	: 1595
00000000G	EF		01	FB	00171	CALLS	#1, REWRITE_RECORD	
			00D2	31	00178	BRW	33\$	: 1473
	53	20	A2	DO	0017B	20\$:	MOVL 32(R2), SMQ_N	: 1606
			53	DD	0017F	PUSHL	SMQ_N	
	6A		01	FB	00181	CALLS	#1, READ_RECORD	
11	A0		04	8A	00184	BICB2	#4, 17(SMQ)	: 1607
			50	DD	00188	PUSHL	SMQ	: 1608
			53	DD	0018A	PUSHL	SMQ_N	
00000000G	EF		02	FB	0018C	CALLS	#2, STOP_SYMBIONT_STREAM	
			76	11	00193	BRB	26\$	: 1609
	54	20	A2	DO	00195	21\$:	MOVL 32(R2), SMQ_N	: 1621

			54	DD	00199	PUSHL	SMQ_N	:	
	6A		01	FB	0019B	CALLS	#1,-READ_RECORD	:	
	53		50	DO	0019E	MOVL	R0, SMQ	:	
10	A3		08	8A	001A1	BICB2	#8, 16(SMQ)	:	1622
			53	DD	001A5	PUSHL	SMQ	:	1623
			54	DD	001A7	PUSHL	SMQ_N	:	
00000000G	EF		02	FB	001A9	CALLS	#2,-PAUSE_EXECUTION	:	
	58		50	DO	001B0	MOVL	R0, STATUS	:	
	2F		58	E9	001B3	BLBC	STATUS, 23\$	:	1624
			38	11	001B6	BRB	24\$	:	1625
	54	20	A2	DO	001B8	MOVL	32(R2), SMQ_N	:	1637
			54	DD	001BC	PUSHL	SMQ_N	:	
	6A		01	FB	001BE	CALLS	#1,-READ_RECORD	:	
10	53		50	DO	001C1	MOVL	R0, SMQ	:	
	A3	40	8F	8A	001C4	BICB2	#64, 16(SMQ)	:	1638
		31	A2	9F	001C9	PUSHAB	49(R2)	:	1642
	7E	30	A2	9A	001CC	MOVZBL	48(R2), -(SP)	:	
	7E	28	A2	7D	001D0	MOVQ	40(R2), -(SP)	:	
		24	A2	DD	001D4	PUSHL	36(R2)	:	
			53	DD	001D7	PUSHL	SMQ	:	
			54	DD	001D9	PUSHL	SMQ_N	:	
00000000G	EF		07	FB	001DB	CALLS	#7,-RESUME_EXECUTION	:	
	58		50	DO	001E2	MOVL	R0, STATUS	:	
			53	DD	001E5	PUSHL	SMQ	:	1643
			54	DD	001E7	PUSHL	SMQ_N	:	
00000000G	EF		02	FB	001E9	CALLS	#2,-FIND_PENDING_JOBS	:	
			FF32	31	001F0	BRW	15\$	:	1644
10	A0		20	8A	001F3	BICB2	#32, 16(SMQ)	:	1656
	53	20	A2	DO	001F7	MOVL	32(R2), SMQ_N	:	1657
			53	DD	001FB	PUSHL	SMQ_N	:	
	6A		01	FB	001FD	CALLS	#1,-READ_RECORD	:	
			50	DO	00200	PUSHL	SMQ	:	1658
			53	DD	00202	PUSHL	SMQ_N	:	
00000000G	EF		02	FB	00204	CALLS	#2,-RESET_EXECUTOR_QUEUE	:	
			53	DD	0020B	PUSHL	SMQ_N	:	1659
00000000G	EF		01	FB	0020D	CALLS	#1,-REWRITE_RECORD	:	
			5B	D4	00214	CLRL	NEXT_ACTION	:	1660
			35	11	00216	BRB	33\$	:	1473
		42	A2	9F	00218	PUSHAB	66(R2)	:	1670
	7E	40	A2	3C	0021B	MOVZWL	64(R2), -(SP)	:	
		20	A2	9F	0021F	PUSHAB	32(R2)	:	1668
		00000000'	EF	9F	00222	PUSHAB	THIS_SYSID	:	1666
00000000G	EF		04	FB	00228	CALLS	#4, BROADCAST_MESSAGE	:	1670
			19	11	0022F	BRB	32\$	:	1671
		20	A2	DD	00231	PUSHL	32(R2)	:	1679
		70	A2	9F	00234	PUSHAB	112(R2)	:	1678
00000000G	EF		02	FB	00237	CALLS	#2, SEND_SERVICE_RESPONSE_MESSAGE	:	
			0A	11	0023E	BRB	32\$	:	1680
		20	A2	DD	00240	PUSHL	32(R2)	:	1686
00000000G	EF		01	FB	00243	CALLS	#1, DELETE_FILES	:	
	5B		01	DO	0024A	MOVL	#1, NEXT_ACTION	:	1687
	50		58	DO	0024D	MOVL	STATUS, R0	:	1695
			04		00250	RET		:	

; Routine Size: 593 bytes, Routine Base: CODE + 0195

```

: 660 1696 1 GLOBAL ROUTINE SCAN_INCOMPLETE_SERVICES(EVENT,P1,P2,P3,P4): NOVALUE=
: 661 1697 1
: 662 1698 1 ++
: 663 1699 1
: 664 1700 1 FUNCTIONAL DESCRIPTION:
: 665 1701 1 This routine scans the incomplete services list when a specified event
: 666 1702 1 that allows an incomplete service to progress has occurred.
: 667 1703 1
: 668 1704 1 INPUT PARAMETERS:
: 669 1705 1 EVENT - Code identifying the event.
: 670 1706 1 P1-P4 - Event-dependent parameters.
: 671 1707 1
: 672 1708 1 IMPLICIT INPUTS:
: 673 1709 1 NONE
: 674 1710 1
: 675 1711 1 OUTPUT PARAMETERS:
: 676 1712 1 NONE
: 677 1713 1
: 678 1714 1 IMPLICIT OUTPUTS:
: 679 1715 1 NONE
: 680 1716 1
: 681 1717 1 ROUTINE VALUE:
: 682 1718 1 NONE
: 683 1719 1
: 684 1720 1 SIDE EFFECTS:
: 685 1721 1 NONE
: 686 1722 1
: 687 1723 1 --
: 688 1724 1
: 689 1725 2 BEGIN
: 690 1726 2 LOCAL
: 691 1727 2 PRED_MODIFIED, ! True if predecessor modified
: 692 1728 2 SRQ_NP, ! Record number of predecessor of SRQ
: 693 1729 2 SRQ_P: REF BBLOCK, ! Pointer to predecessor of SRQ
: 694 1730 2 SRQ_N; ! Record number of SRQ
: 695 1731 2
: 696 1732 2
: 697 1733 2 ! Search the incomplete service list for those that are affected by the
: 698 1734 2 ! specified event and process these.
: 699 1735 2
: 700 1736 2 PRED_MODIFIED = FALSE;
: 701 1737 2 SRQ_P = READ_RECORD(SRQ_NP = SQH$K_RECNO);
: 702 1738 2 SRQ_N = .SRQ_P[SQH$L_INCOMPLETE_SERVICE_LIST];
: 703 1739 2 WHILE .SRQ_N.NEQ 0 DO
: 704 1740 2 BEGIN
: 705 1741 2 LOCAL
: 706 1742 2 SRQ: REF BBLOCK, ! Pointer to SRQ
: 707 1743 2 SRQ_NS, ! Record number of successor of SRQ
: 708 1744 2 STATUS, ! Request status
: 709 1745 2 NEXT_ACTION; ! Code for next action
: 710 1746 2
: 711 1747 2
: 712 1748 2 SRQ = READ_RECORD(.SRQ_N);
: 713 1749 2 SRQ_NS = .SRQ[SYMS$L_LINK];
: 714 1750 2
: 715 1751 2
: 716 1752 3 ! Check for corrupted incomplete services list. If an incorrect record type
```

```

: 717 1753 3 ! is found, truncate the list. The remaining records are either already
: 718 1754 3 ! linked to another list, or they will be lost until a cold start operation
: 719 1755 3 ! is performed. Pruning these unwanted records (most likely free list or
: 720 1756 3 ! job header records) from the incomplete services list will prevent
: 721 1757 3 ! reading them every time SCAN_INCOMPLETE_SERVICES is called.
: 722 1758 3
: 723 1759 3 IF .SRQ[SYMSB_TYPE] NEQ SYMSK_SRQ
: 724 1760 3 THEN
: 725 1761 4 BEGIN
: 726 1762 4   DIAG_TRACE[12] = .DIAG_TRACE[12] + 1;
: 727 1763 4   DIAG_TRACE[13] = .SRQ[SYMSB_TYPE] * 65536 + .SRQ_N;
: 728 1764 4   IF .FLAGS[FLAGS_V LOG OF REPAIR]
: 729 1765 4     THEN SIGNAL(JBCS_DIAGNOSTIC OR STSK_INFO, 1,
: 730 1766 4       $DESCRIPTOR('on-line repair of incomplete services list')) ;
: 731 1767 4   IF .SRQ NP EQL SQH$K_RECNO
: 732 1768 4     THEN SRQ_P[SQH$L_INCOMPLETE_SERVICE_LIST] = 0
: 733 1769 4     ELSE SRQ_P[SYMS$L_LINK] = 0;
: 734 1770 4   PRED_MODIFIED = TRUE;
: 735 1771 4   EXIT[LOOP];
: 736 1772 3 END;
: 737 1773 3
: 738 1774 3
: 739 1775 3 STATUS = SS$ NORMAL;
: 740 1776 3 NEXT_ACTION = K_RELEASE;
: 741 1777 3
: 742 1778 3
: 743 1779 3 CASE .EVENT FROM ISRV_K_REMOTE TO ISRV_K_PURGE_SJH OF
: 744 1780 3 SET
: 745 1781 3
: 746 1782 3   [ISRV_K_REMOTE]:
: 747 1783 3     BEGIN
: 748 1784 4     IF SYSID_EQL(THIS_SYSID, SRQ[SRQ$T_RECEIVING_SYSID])
: 749 1785 5     AND NOT .SRQ[SRQ$V_STALLED]
: 750 1786 4     THEN
: 751 1787 4       STATUS = PROCESS_REMOTE_SERVICES(.SRQ; NEXT_ACTION);
: 752 1788 4     END;
: 753 1789 3
: 754 1790 3   [ISRV_K_SYNCHRONIZE]:
: 755 1791 3     BEGIN
: 756 1792 3     BIND
: 757 1793 4       SJH_N          = P1,          ! Record number of SJH
: 758 1794 4       STS          = P2;          ! Completion status
: 759 1795 4
: 760 1796 4     IF .SRQ[SRQ$L_FUNCTION_CODE] EQL SRQ$K_SYNCHRONIZE_JOB
: 761 1797 4     AND .SRQ[SRQ$[_P1]] EQL .SJH_N
: 762 1798 4     THEN
: 763 1799 4       BEGIN
: 764 1800 4       NEXT_ACTION = K_COMPLETE;
: 765 1801 5       STATUS = .STS;
: 766 1802 5     END;
: 767 1803 5   END;
: 768 1804 4
: 769 1805 3
: 770 1806 3   [ISRV_K_SYMBIONT]:
: 771 1807 3     BEGIN
: 772 1808 3
: 773 1809 4
```

```

774 1810 4 BIND
775 1811 4 SMQ_N = P1, ! Record number of SMQ
776 1812 4 SMQ = P2, ! Pointer to SMQ
777 1813 4 FUNC = P3, ! Function completed
778 1814 4 STS = P4; ! Completion status
779 1815 4
780 1816 4 IF .SRQ[SRQ$L_FUNCTION_CODE] EQL SRQ$K_START_SYMBIONT
781 1817 4 AND .SRQ[SRQ$_P1] EQL .SMQ_N
782 1818 5 AND (.FUNC EQL 0 OR .FUNC EQL .SRQ[SRQ$L_FUNCTION_CODE])
783 1819 4 THEN
784 1820 5 BEGIN
785 1821 5 NEXT_ACTION = K_COMPLETE;
786 1822 5 STATUS = .STS;
787 1823 4 END;
788 1824 3 END;
789 1825 3
790 1826 3
791 1827 3 [ISRV_K_PURGE_SYSID]:
792 1828 4 BEGIN
793 1829 4 BIND
794 1830 4 SYSID = P1; ! Pointer to system ID
795 1831 4
796 1832 5 IF SYSID_EQL(.SYSID, SRQ[SRQ$T_SENDING_SYSID])
797 1833 4 THEN
798 1834 4 NEXT_ACTION = K_DEALLOCATE
799 1835 4
800 1836 5 ELSE IF SYSID_EQL(.SYSID, SRQ[SRQ$T_RECEIVING_SYSID])
801 1837 4 THEN
802 1838 5 BEGIN
803 1839 5 STATUS = JBC$_SYSFAIL OR STS$K_ERROR;
804 1840 5 NEXT_ACTION = K_COMPLETE;
805 1841 4 END;
806 1842 3 END;
807 1843 3
808 1844 3
809 1845 3 [ISRV_K_PURGE_SMQ]:
810 1846 4 BEGIN
811 1847 4 BIND
812 1848 4 SMQ_N = P1; ! Record number of SMQ
813 1849 4
814 1850 4 IF
815 1851 4 P ONEOF (.SRQ[SRQ$L_FUNCTION_CODE], BMSK_(
816 1852 4 P SRQ$K_START_QUEUE,
817 1853 4 P SRQ$K_STOP_QUEUE,
818 1854 4 P SRQ$K_PAUSE_QUEUE,
819 1855 4 P SRQ$K_RESUME_QUEUE,
820 1856 4 P SRQ$K_RESET_QUEUE,
821 1857 5 SRQ$K_START_SYMBIONT))
822 1858 4 AND .SRQ[SRQ$_P1] EQL .SMQ_N
823 1859 4 THEN
824 1860 4 NEXT_ACTION = K_COMPLETE;
825 1861 3 END;
826 1862 3
827 1863 3
828 1864 3 [ISRV_K_PURGE_SJH]:
829 1865 4 BEGIN
830 1866 4 BIND

```

```

: 831      1867 4      SJH_N      = P1;      ! Record number of SJH
: 832      1868 4
: 833      1869 4
: 834      1870 4      P      IF      ONEOF (,SRQ[SRQ$L_FUNCTION_CODE], BMSK_(
: 835      1871 4      P      SRQ$K_START_JOB,
: 836      1872 5      SRQ$K_ABORT_JOB))
: 837      1873 4      AND .SRQ[SRQ$L_P2] EQL .SJH_N
: 838      1874 4      THEN
: 839      1875 4      NEXT_ACTION = K_COMPLETE;
: 840      1876 3      END;
: 841      1877 3
: 842      1878 3
: 843      1879 3      TES;
: 844      1880 3
: 845      1881 3
: 846      1882 3      IF .NEXT_ACTION EQL K_COMPLETE
: 847      1883 3      THEN
: 848      1884 4      BEGIN
: 849      1885 4
: 850      1886 4      ! If no response is required, merely deallocate the SRQ.
: 851      1887 4      !
: 852      1888 4      IF .SRQ[SRQ$V_NO_RESPONSE]
: 853      1889 4      THEN
: 854      1890 4      NEXT_ACTION = K_DEALLOCATE
: 855      1891 4
: 856      1892 4
: 857      1893 4      ! If the response can be sent locally, send it and deallocate the SRQ.
: 858      1894 4      !
: 859      1895 5      ELSE IF SYSID_EQL(THIS_SYSID, SRQ[SRQ$T_SENDING_SYSID])
: 860      1896 4      THEN
: 861      1897 5      BEGIN
: 862      1898 5      SEND_SERVICE_RESPONSE_MESSAGE(SRQ[SRQ$T_SRB], .STATUS);
: 863      1899 5      NEXT_ACTION = K_DEALLOCATE;
: 864      1900 5      END
: 865      1901 5
: 866      1902 5
: 867      1903 5      ! Otherwise, convert the SRQ to a "response" request and forward it
: 868      1904 5      ! to the sending job controller.
: 869      1905 5      !
: 870      1906 4      ELSE
: 871      1907 5      BEGIN
: 872      1908 5      COPY_SYSID(SRQ[SRQ$T_SENDING_SYSID], SRQ[SRQ$T_RECEIVING_SYSID]);
: 873      1909 5      COPY_SYSID(THIS_SYSID, SRQ[SRQ$T_SENDING_SYSID]);
: 874      1910 5      SRQ[SRQ$L_FUNCTION_CODE] = SRQ$K_RESPONSE;
: 875      1911 5      SRQ[SRQ$L_P1] = .STATUS;
: 876      1912 5      SRQ[SRQ$V_STALLED] = FALSE;
: 877      1913 5      ENTER_REMOTE_REQUEST(SRQ[SRQ$T_RECEIVING_SYSID]);
: 878      1914 5      NEXT_ACTION = K_REWRITE;
: 879      1915 4      END;
: 880      1916 3      END;
: 881      1917 3
: 882      1918 3
: 883      1919 3      CASE .NEXT_ACTION FROM K_DEALLOCATE TO K_REWRITE OF
: 884      1920 3      SET
: 885      1921 3
: 886      1922 3
: 887      1923 3      [K_DEALLOCATE]:

```

```

: 888      1924 4      BEGIN
: 889      1925 4      IF .SRQ_NP EQL SQH$K_RECNO
: 890      1926 4      THEN SRQ_P[SQH$K_INCOMPLETE_SERVICE_LIST] = .SRQ_NS
: 891      1927 4      ELSE SRQ_P[SYMSL_LINK] = .SRQ_NS;
: 892      1928 4
: 893      1929 4      ! First, rewrite the predecessor, then deallocate the SRQ.
: 894      1930 4      ! If done in the opposite order, a crash after the deallocate
: 895      1931 4      ! can result in a corrupted INCOMPLETE_SERVICE_LIST, which
: 896      1932 4      ! will then result in a queue format error on warm/cold start.
: 897      1933 4
: 898      1934 4      FLUSH_RECORD(.SRQ_NP);
: 899      1935 4      DEALLOCATE_RECORD(.SRQ_N);
: 900      1936 3      END;
: 901      1937 3
: 902      1938 3
: 903      1939 3      [K_RELEASE]:
: 904      1940 4      BEGIN
: 905      1941 4      IF TESTBITSC(PRED_MODIFIED)
: 906      1942 4      THEN REWRITE_RECORD(.SRQ_NP)
: 907      1943 4      ELSE RELEASE_RECORD(.SRQ_NP);
: 908      1944 4      SRQ_NP = .SRQ_N;
: 909      1945 4      SRQ_P = .SRQ;
: 910      1946 3      END;
: 911      1947 3
: 912      1948 3
: 913      1949 3      [K_REWRITE]:
: 914      1950 4      BEGIN
: 915      1951 4      IF TESTBITSS(PRED_MODIFIED)
: 916      1952 4      THEN REWRITE_RECORD(.SRQ_NP)
: 917      1953 4      ELSE RELEASE_RECORD(.SRQ_NP);
: 918      1954 4      SRQ_NP = .SRQ_N;
: 919      1955 4      SRQ_P = .SRQ;
: 920      1956 3      END;
: 921      1957 3
: 922      1958 3
: 923      1959 3      TES;
: 924      1960 3
: 925      1961 3
: 926      1962 3      SRQ_N = .SRQ_NS;
: 927      1963 2      END;
: 928      1964 2
: 929      1965 2
: 930      1966 2      IF .PRED_MODIFIED
: 931      1967 2      THEN REWRITE_RECORD(.SRQ_NP)
: 932      1968 2      ELSE RELEASE_RECORD(.SRQ_NP);
: 933      1969 1      END;

```

```

20 72 69 61 70 65 72 20 65 6E 69 6C 2D 6E 6F 003E6 P.AAB: .ASCII \on-line repair of incomplete services li\
73 20 65 74 65 6C 70 6D 6F 63 6E 69 20 66 6F 003F5
      69 6C 20 73 65 63 69 76 72 65 00404
      74 73 0040E
      0000002A 00410 P.AAA: .ASCII \st\
      00000000 00414 .LONG 42
      .ADDRESS P.AAB

```

				OFFC 00000	.ENTRY	SCAN_INCOMPLETE_SERVICES, Save R2,R3,R4,R5,-;	1696
		5A	00000000G	EF 9E 00002	MOVAB	R6,R7,R8,R9,R10,R11	
		59	000000000	EF 9E 00009	MOVAB	REWRITE_RECORD, R10	
				57 D4 00010	CLRL	THIS_SYSID, R9	1736
		55		01 D0 00012	MOVL	PRED_MODIFIED	1737
				01 DD 00015	PUSHL	#1, SRQ_NP	
			00000000G	01 FB 00017	CALLS	#1, READ_RECORD	
		53		50 D0 0001E	MOVL	R0, SRQ_P	
		56	44	A3 D0 00021	MOVL	68(SRQ_P), SRQ_N	1738
				4D 13 00025	BEQL	5\$	1739
				56 DD 00027	PUSHL	SRQ_N	1748
			00000000G	01 FB 00029	CALLS	#1, READ_RECORD	
		52		50 D0 00030	MOVL	R0, SRQ	
		58		62 D0 00033	MOVL	(SRQ), SRQ_NS	1749
		09	04	A2 91 00036	CMPB	4(SRQ), #9	1759
				3B 13 0003A	BEQL	6\$	
			FE28	C9 D6 0003C	INCL	DIAG_TRACE+48	1762
		50	04	A2 9A 00040	MOVZBL	4(SRQ), R0	1763
		50		10 78 00044	ASHL	#16, R0, R0	
FE2C	50	50		56 C1 00048	ADDL3	SRQ_N, R0, DIAG_TRACE+52	
	C9	50		05 E1 0004E	BBC	#5, FLAGS+2, 2\$	1764
	12	FA	A9	AF 9F 00053	PUSHAB	P.AAA	1766
			A2	01 DD 00056	PUSHL	#1	1765
				8F DD 00058	PUSHL	#296115	
			00000000G	03 FB 0005E	CALLS	#3, LIB\$SIGNAL	
		00		55 D1 00065	CMPL	SRQ_NP, #1	1767
		01		05 12 00068	BNEQ	3\$	
			44	A3 D4 0006A	CLRL	68(SRQ_P)	1768
				02 11 0006D	BRB	4\$	
				63 D4 0006F	CLRL	(SRQ_P)	1769
		57		01 D0 00071	MOVL	#1, PRED_MODIFIED	1770
				015F 31 00074	BRW	34\$	1761
		54		01 D0 00077	MOVL	#1, STATUS	1775
		5B		02 D0 0007A	MOVL	#2, NEXT_ACTION	1776
		00		AC CF 0007D	CASEL	EVENT, #0, #5	1779
0060	05		04	000C 00082	.WORD	8\$-7\$,-	
	003F	002A		008C 0008A		9\$-7\$,-	
		009E				10\$-7\$,-	
						12\$-7\$,-	
						16\$-7\$,-	
						17\$-7\$	
		1A	A2	69 D1 0008E	CMPL	THIS_SYSID, 26(SRQ)	1785
				6F 12 00092	BNEQ	15\$	
		1E	A2	A9 B1 00094	CMPW	THIS_SYSID+4, 30(SRQ)	
			04	68 12 00099	BNEQ	15\$	
		56	10	01 E0 0009B	BBS	#1, 16(SRQ), 13\$	1786
				52 DD 000A0	PUSHL	SRQ	1788
		FCD6	CF	01 FB 000A2	CALLS	#1, PROCESS_REMOTE_SERVICES	
			54	50 D0 000A7	MOVL	R0, STATUS	
				4A 11 000AA	BRB	13\$	1779
			03	A2 D1 000AC	CMPL	12(SRQ), #3	1798
				7E 12 000B0	BNEQ	18\$	
		08	AC	A2 D1 000B2	CMPL	32(SRQ), SJH_N	1799
			20	7B 12 000B7	BNEQ	20\$	
				5B D4 000B9	CLRL	NEXT_ACTION	1802

	54	0C	AC	D0	000BB	MOVL	STS, STATUS	1803	
			73	11	000BF	BRB	20\$	1779	
	0C	0C	A2	D1	000C1	10\$: CMPL	12(SRQ), #12	1816	
			6D	12	000C5	BNEQ	20\$		
08	AC	20	A2	D1	000C7	CMPL	32(SRQ), SMQ_N	1817	
			66	12	000CC	BNEQ	20\$		
		10	AC	D5	000CE	TSTL	FUNC	1818	
			07	13	000D1	BEQL	11\$		
0C	A2	10	AC	D1	000D3	CMPL	FUNC, 12(SRQ)		
			5A	12	000D8	BNEQ	20\$		
			5B	D4	000DA	11\$: CLRL	NEXT_ACTION	1821	
	54	14	AC	D0	000DC	MOVL	STS, STATUS	1822	
			52	11	000E0	BRB	20\$	1779	
	50	08	AC	D0	000E2	12\$: MOVL	SYSID, R0	1832	
14	A2		60	D1	000E6	CMPL	(R0), 20(SRQ)		
			0C	12	000EA	BNEQ	14\$		
18	A2	04	A0	B1	000EC	CMPW	4(R0), 24(SRQ)		
			05	12	000F1	BNEQ	14\$		
	5B		01	D0	000F3	MOVL	#1, NEXT_ACTION	1834	
			3C	11	000F6	13\$: BRB	20\$		
1A	A2		60	D1	000F8	14\$: CMPL	(R0), 26(SRQ)	1836	
			36	12	000FC	BNEQ	20\$		
1E	A2	04	A0	B1	000FE	CMPW	4(R0), 30(SRQ)		
			2F	12	00103	15\$: BNEQ	20\$		
	54	000480F2	8F	D0	00105	MOVL	#295154, STATUS	1839	
			24	11	0010C	BRB	19\$	1840	
50	0F880000	8F	0C	A2	78	0010E	16\$: ASHL	12(SRQ), #260571136, R0	1857
			1B	18	00117	BGEQ	20\$		
	08	AC	20	A2	D1	00119	CMPL	32(SRQ), SMQ_N	1858
			10	11	0011E	BRB	18\$		
50	60000000	8F	0C	A2	78	00120	17\$: ASHL	12(SRQ), #1610612736, R0	1872
			09	18	00129	BGEQ	20\$		
	08	AC	24	A2	D1	0012B	CMPL	36(SRQ), SJH_N	1873
			02	12	00130	18\$: BNEQ	20\$		
			5B	D4	00132	19\$: CLRL	NEXT_ACTION	1875	
			5B	D5	00134	20\$: TSTL	NEXT_ACTION	1882	
			4C	12	00136	BNEQ	23\$		
	19	10	A2	E8	00138	BLBS	16(SRQ), 21\$	1888	
14	A2		69	D1	0013C	CMPL	THIS_SYSID, 20(SRQ)	1895	
			18	12	00140	BNEQ	22\$		
18	A2	04	A9	B1	00142	CMPW	THIS_SYSID+4, 24(SRQ)		
			11	12	00147	BNEQ	22\$		
			54	DD	00149	PUSHL	STATUS	1898	
		70	A2	9F	0014B	PUSHAB	112(SRQ)		
00000000G	EF		02	FB	0014E	CALLS	#2, SEND_SERVICE_RESPONSE_MESSAGE		
	5B		01	D0	00155	21\$: MOVL	#1, NEXT_ACTION	1899	
			2A	11	00158	BRB	23\$	1895	
	1A	A2	14	A2	D0	0015A	22\$: MOVL	20(SRQ), 26(SRQ)	1908
	1E	A2	18	A2	B0	0015F	MOVW	24(SRQ), 30(SRQ)	
	14	A2		69	D0	00164	MOVL	THIS_SYSID, 20(SRQ)	1909
	18	A2	04	A9	B0	00168	MOVW	THIS_SYSID+4, 24(SRQ)	
	0C	A2		0A	D0	0016D	MOVL	#10, 12(SRQ)	1910
	20	A2		54	D0	00171	MOVL	STATUS, 32(SRQ)	1911
	10	A2		02	8A	00175	BICB2	#2, 16(SRQ)	1912
		1A	A2	9F	00179	PUSHAB	26(SRQ)	1913	
0000V	CF		01	FB	0017C	CALLS	#1, ENTER_REMOTE_REQUEST		
	5B		03	D0	00181	MOVL	#3, NEXT_ACTION	1914	

02	01	5B	CF	00184	23\$:	CASEL	NEXT_ACTION, #1, #2	:	1919
002E	0028	0006		00188	24\$:	.WORD	25\$-24\$,-	:	
							28\$-24\$,-	:	
							29\$-24\$	:	
	01	55	D1	0018E	25\$:	CMPL	SRQ_NP, #1	:	1925
		06	12	00191		BNEQ	26\$	:	
	44	58	D0	00193		MOVL	SRQ_NS, 68(SRQ_P)	:	1926
		03	11	00197		BRB	27\$	:	
	63	58	D0	00199	26\$:	MOVL	SRQ_NS, (SRQ_P)	:	1927
		55	DD	0019C	27\$:	PUSHL	SRQ_NP	:	1934
00000000G	EF	01	FB	0019E		CALLS	#1, -FLUSH_RECORD	:	
		56	DD	001A5		PUSHL	SRQ_N	:	1935
00000000G	EF	01	FB	001A7		CALLS	#1, -DEALLOCATE_RECORD	:	
		20	11	001AE		BRB	33\$	:	1919
0D	57	00	E5	001B0	28\$:	BBCC	#0, PRED_MODIFIED, 31\$	:	1941
		04	11	001B4		BRB	30\$	:	1942
07	57	00	E3	001B6	29\$:	BBCS	#0, PRED_MODIFIED, 31\$	:	1951
		55	DD	001BA	30\$:	PUSHL	SRQ_NP	:	1952
	6A	01	FB	001BC		CALLS	#1, -REWRITE_RECORD	:	
		09	11	001BF		BRB	32\$	:	
		55	DD	001C1	31\$:	PUSHL	SRQ_NP	:	1953
00000000G	EF	01	FB	001C3		CALLS	#1, -RELEASE_RECORD	:	
	55	56	D0	001CA	32\$:	MOVL	SRQ_N, SRQ_NP	:	1954
	53	52	D0	001CD		MOVL	SRQ, SRQ_P	:	1955
	56	58	D0	001D0	33\$:	MOVL	SRQ_NS, SRQ_N	:	1962
		FE4F	31	001D3		BRW	1\$	:	1739
	06	57	E9	001D6	34\$:	BLBC	PRED_MODIFIED, 35\$	:	1966
		55	DD	001D9		PUSHL	SRQ_NP	:	1967
	6A	01	FB	001DB		CALLS	#1, -REWRITE_RECORD	:	
			04	001DE		RET		:	
		55	DD	001DF	35\$:	PUSHL	SRQ_NP	:	1968
00000000G	EF	01	FB	001E1		CALLS	#1, -RELEASE_RECORD	:	
		04	001E8		RET			:	1969

; Routine Size: 489 bytes, Routine Base: CODE + 0418

```

: 935      1970 1 GLOBAL ROUTINE REMOTE_BLOCKING_AST: NOVALUE=
: 936      1971 1
: 937      1972 1 |++
: 938      1973 1
: 939      1974 1 | FUNCTIONAL DESCRIPTION:
: 940      1975 1 |     This routine is the blocking AST routine for the job controller remote
: 941      1976 1 |     request lock. This routine is entered when another job controller
: 942      1977 1 |     attempts to obtain this job controller's remote request lock.
: 943      1978 1
: 944      1979 1 | INPUT PARAMETERS:
: 945      1980 1 |     Standard AST routine parameters (not used).
: 946      1981 1
: 947      1982 1 | IMPLICIT INPUTS:
: 948      1983 1 |     NONE
: 949      1984 1
: 950      1985 1 | OUTPUT PARAMETERS:
: 951      1986 1 |     NONE
: 952      1987 1
: 953      1988 1 | IMPLICIT OUTPUTS:
: 954      1989 1 |     NONE
: 955      1990 1
: 956      1991 1 | ROUTINE VALUE:
: 957      1992 1 |     NONE
: 958      1993 1
: 959      1994 1 | SIDE EFFECTS:
: 960      1995 1 |     NONE
: 961      1996 1
: 962      1997 1 | --
: 963      1998 1
: 964      1999 2 BEGIN
: 965      2000 2 LOCAL
: 966      2001 2     STATUS_1,           ! Status return
: 967      2002 2     STATUS_2;         ! Status return
: 968      2003 2
: 969      2004 2
: 970      2005 2 ! Convert the lock to null mode to allow the process that has requested the
: 971      2006 2 ! lock to obtain it.
: 972      2007 2
: 973      2008 2 STATUS 1 = $ENQW(
: 974      2009 2     EFN=JBC$K_SYNC_EFN,
: 975      2010 2     LKMODE=LCK$K_NCMODE,
: 976      2011 2     LKSB=REMOTE_REQUEST_LKSB,
: 977      2012 2     FLAGS=LCK$M_CONVERT);
: 978      2013 2 IF .STATUS_1 THEN STATUS_1 = .REMOTE_REQUEST_LKSB[0];
: 979      2014 2 IF NOT .STATUS_1
: 980      2015 2 THEN
: 981      2016 2     SIGNAL(JBC$_COMREMJBC OR STS$K_ERROR, 0, .STATUS_1);
: 982      2017 2
: 983      2018 2
: 984      2019 2 ! Reconvert the lock to exclusive mode to reenale the blocking AST.
: 985      2020 2
: 986      2021 2 STATUS 2 = $ENQ(
: 987      2022 2     LKMODE=LCK$K_EXMODE,
: 988      2023 2     LKSB=REMOTE_REQUEST_LKSB,
: 989      2024 2     FLAGS=LCK$M_CONVERT OR LCK$M_NODLCKBLK,
: 990      2025 2     ASTADR=REMOTE_COMPLETION_AST,
: 991      2026 2     BLKAST=REMOTE_BLOCKING_AST);
```

```

.: 992      2027 2 IF NOT .STATUS_2
.: 993      2028 2 THEN
.: 994      2029 2     SIGNAL(JBC$_COMREMJBC OR STS$_K_ERROR, 0, .STATUS_2);
.: 995      2030 1 END;

```

Address	Hex	ASCII	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	
---------	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--

; Routine Size: 107 bytes, Routine Base: CODE + 0601

```

: 997      2031 1 ROUTINE REMOTE_COMPLETION_NONAST: NOVALUE=
: 998      2032 1
: 999      2033 1 ++
1000      2034 1
1001      2035 1 FUNCTIONAL DESCRIPTION:
1002      2036 1 This routine is scheduled to execute by the completion AST routine for
1003      2037 1 reconversion of the job controller remote request lock to exclusive
1004      2038 1 mode, which is entered when another job controller has obtained and
1005      2039 1 released this job controller's remote request lock.
1006      2040 1
1007      2041 1 INPUT PARAMETERS:
1008      2042 1 NONE
1009      2043 1
1010      2044 1 IMPLICIT INPUTS:
1011      2045 1 NONE
1012      2046 1
1013      2047 1 OUTPUT PARAMETERS:
1014      2048 1 NONE
1015      2049 1
1016      2050 1 IMPLICIT OUTPUTS:
1017      2051 1 NONE
1018      2052 1
1019      2053 1 ROUTINE VALUE:
1020      2054 1 NONE
1021      2055 1
1022      2056 1 SIDE EFFECTS:
1023      2057 1 NONE
1024      2058 1
1025      2059 1 --
1026      2060 1
1027      2061 2 BEGIN
1028      2062 2
1029      2063 2 ! Get the current time.
1030      2064 2
1031      2065 2 $GETTIM(TIMADR=CUR_TIME);
1032      2066 2
1033      2067 2
1034      2068 2 IF .QUEUE_FAB[FAB$W_IF1] NEQ 0
1035      2069 2 THEN
1036      2070 2 BEGIN
1037      2071 2
1038      2072 2 ! Lock the queue file.
1039      2073 2
1040      2074 2 LOCK_QUEUE_FILE();
1041      2075 2
1042      2076 2
1043      2077 2 ! Search the incomplete services list to perform actions requested by the
1044      2078 2 remote job controller.
1045      2079 2
1046      2080 2 SCAN_INCOMPLETE_SERVICES(ISRV_K_REMOTE);
1047      2081 2
1048      2082 2
1049      2083 2 ! Unlock the queue file.
1050      2084 2
1051      2085 2 UNLOCK_QUEUE_FILE();
1052      2086 2 END;
: 1053      2087 1 END;
```

```

                                .EXTRN  SYS$GETTIM
                                0000 00000 REMOTE_COMPLETION_NONAST:
                                .WORD  Save nothing
00000000G 00 00000000' EF 9F 00002  PUSHAB  CUR_TIME          : 2031
                                01 FB 00008  CALLS  #1, SYS$GETTIM      : 2065
                                00000000' EF B5 0000F  TSTW  QUEUE_FAB+2    : 2068
                                15 13 00015  BEQL  1$
00000000G EF 00 FB 00017  CALLS  #0, LOCK_QUEUE_FILE      : 2074
                                7E D4 0001E  CLRL  -(SP)         : 2080
                                01 FB 00020  CALLS  #1, SCAN_INCOMPLETE_SERVICES
                                00 FB 00025  CALLS  #0, UNLOCK_QUEUE_FICE
                                04 0002C 1$:  RET                : 2085
                                                                : 2087

```

; Routine Size: 45 bytes, Routine Base: CODE + 066C

```
2088 1 ROUTINE REMOTE_COMPLETION_AST: NOVALUE=
2089 1
2090 1 ++
2091 1
2092 1 FUNCTIONAL DESCRIPTION:
2093 1 This routine is the completion AST routine for reconversion of the job
2094 1 controller remote request lock to exclusive mode. This routine is
2095 1 entered when another job controller has obtained and released this job
2096 1 controller's remote request lock.
2097 1
2098 1 INPUT PARAMETERS:
2099 1 Standard AST routine parameters (not used).
2100 1
2101 1 IMPLICIT INPUTS:
2102 1 NONE
2103 1
2104 1 OUTPUT PARAMETERS:
2105 1 NONE
2106 1
2107 1 IMPLICIT OUTPUTS:
2108 1 NONE
2109 1
2110 1 ROUTINE VALUE:
2111 1 NONE
2112 1
2113 1 SIDE EFFECTS:
2114 1 NONE
2115 1
2116 1 --
2117 1
2118 2 BEGIN
2119 2
2120 2 ! Check status of the $ENQ.
2121 2
2122 2 IF NOT .REMOTE_REQUEST_LKSB[0]
2123 2 THEN
2124 2 SIGNAL(JBC$_COMREMJBC OR STS$K_ERROR, 0, .REMOTE_REQUEST_LKSB[0]);
2125 2
2126 2
2127 2 ! Schedule the companion routine to execute.
2128 2
2129 2 SCHEDULE_NONAST(REMOTE_COMPLETION_NONAST);
2130 1 END;
```

0004 00000 REMOTE_COMPLETION_AST:					
			.WORD	Save R2	: 2088
52	00000000'	EF 9E 00002	MOVAB	REMOTE_REQUEST_LKSB, R2	: 2122
12		62 E8 00009	BLBS	REMOTE_REQUEST_LKSB, 1\$	: 2124
7E		62 3C 0000C	MOVZWL	REMOTE_REQUEST_LKSB, -(SP)	
		7E D4 0000F	CLRL	-(SP)	
	00048412	8F DD 00011	PUSHL	#295954	
00000000G 00		03 FB 00017	CALLS	#3, LIB\$SIGNAL	
	B2	AF 9F 0001E 1\$:	PUSHAB	REMOTE_COMPLETION_NONAST	: 2129

ASYNCHRON  
V04-002

Asynchronous service management

K 6  
15-Sep-1984 23:49:14  
14-Sep-1984 22:32:32

VAX-11 Bliss-32 V4.0-742  
[JOBCTL.SRC]ASYNCHRON.B32;3

Page 37  
(8)

00000000G EF

01 FB 00021  
04 00028

CALLS #1, SCHEDULE\_NONAST  
RET

: 2130

; Routine Size: 41 bytes, Routine Base: CODE + 0699

BA  
VO

```

: 1099 2131 1 ROUTINE ENTER_REMOTE_REQUEST(SYSID): NOVALUE=
: 1100 2132 1
: 1101 2133 1 ++
: 1102 2134 1
: 1103 2135 1 FUNCTIONAL DESCRIPTION:
: 1104 2136 1 This routine requests services of another job controller.
: 1105 2137 1
: 1106 2138 1 INPUT PARAMETERS:
: 1107 2139 1 SYSID - Address of the system ID of the target.
: 1108 2140 1
: 1109 2141 1 IMPLICIT INPUTS:
: 1110 2142 1 NONE
: 1111 2143 1
: 1112 2144 1 OUTPUT PARAMETERS:
: 1113 2145 1 NONE
: 1114 2146 1
: 1115 2147 1 IMPLICIT OUTPUTS:
: 1116 2148 1 NONE
: 1117 2149 1
: 1118 2150 1 ROUTINE VALUE:
: 1119 2151 1 NONE
: 1120 2152 1
: 1121 2153 1 SIDE EFFECTS:
: 1122 2154 1 NONE
: 1123 2155 1
: 1124 2156 1 --
: 1125 2157 1
: 1126 2158 2 BEGIN
: 1127 2159 2 MAP
: 1128 2160 2 SYSID: REF BBLOCK; ! Pointer to system ID
: 1129 2161 2 LOCAL
: 1130 2162 2 LKSB: REF BBLOCK, ! Pointer to LKSB from dynamic memory
: 1131 2163 2 RESNAM_DESC: VECTOR[2], ! Descriptor for resource name
: 1132 2164 2 RESNAM: BBLOCK[10], ! Buffer for resource name
: 1133 2165 2 STATUS; ! Status return
: 1134 2166 2
: 1135 2167 2
: 1136 2168 2 ! Allocate and initialize the LKSB.
: 1137 2169 2
: 1138 2170 2 LKSB = ALLOCATE_MEMORY();
: 1139 2171 2
: 1140 2172 2
: 1141 2173 2 ! Initialize the resource name.
: 1142 2174 2
: 1143 2175 2 RESNAM[0,0,32,0] = 'JBC$';
: 1144 2176 2 COPY SYSID(SYSID, RESNAM[4,0,0,0]);
: 1145 2177 2 RESNAM_DESC[0] = %ALLOCATION(RESNAM);
: 1146 2178 2 RESNAM_DESC[1] = RESNAM;
: 1147 2179 2
: 1148 2180 2
: 1149 2181 2 ! Enqueue for the doorbell lock of the remote system.
: 1150 2182 2
: 1151 P 2183 2 STATUS = $ENQ(
: 1152 P 2184 2 LKMODE=LCK$K_EXMODE,
: 1153 P 2185 2 LKSB=.LKSB,
: 1154 P 2186 2 RESNAM=RESNAM_DESC,
: 1155 P 2187 2 FLAGS=LCK$M_SYNCSTS OR LCK$M_NODLCKWT,

```

```

: 1156 P 2188 2 ASTADR=ENTER_REMOTE_REQUEST_AST,
: 1157 2189 2 ASTPRM=.LKSB;
: 1158 2190 2
: 1159 2191 2
: 1160 2192 2 ! Set flag is there is no doorbell lock defined for the remote job controller.
: 1161 2193 2 ! This indicates that either the remote node is not available (or not in the
: 1162 2194 2 ! cluster) or the remote job controller does not have the queue file open).
: 1163 2195 2 ! In either case the remote job controller may not respond for a long time or
: 1164 2196 2 ! possibly never (if an invalid node name were specified, for example).
: 1165 2197 2
: 1166 2198 2 ! Note that if SS$_SYNCH is set, the AST will not be delivered. Consequently,
: 1167 2199 2 ! lock dequeuing and memory deallocation must be performed here when the AST
: 1168 2200 2 ! routine is not executed.
: 1169 2201 2
: 1170 2202 2 FLAG$[FLAG$ V NO_REMOTE_DOORBELL] = FALSE;
: 1171 2203 2 IF .STATUS EQC SS$_SYNCH
: 1172 2204 2 THEN
: 1173 2205 2 BEGIN
: 1174 2206 2 FLAG$[FLAG$ V NO_REMOTE_DOORBELL] = TRUE;
: 1175 2207 2 $DEQ(LKID=.LKSB[4,0,32,0]);
: 1176 2208 2 DEALLOCATE_MEMORY(.LKSB);
: 1177 2209 2 END;
: 1178 2210 2
: 1179 2211 2
: 1180 2212 2 ! Check for service failure.
: 1181 2213 2
: 1182 2214 2 IF NOT .STATUS
: 1183 2215 2 THEN
: 1184 2216 2 SIGNAL(JBC$_COMREMJBC OR STS$_ERROR, 0, .STATUS);
: 1185 2217 1 END;

```

.EXTRN SYSS\$DEQ

				001C 00000 ENTER_REMOTE_REQUEST:		
	54	00000000'	EF 9E 00002	.WORD	Save R2,R3,R4	: 2131
	5E		14 C2 00009	MOVAB	FLAGS, R4	
00000000G	EF		00 FB 0000C	SUBL2	#20, SP	: 2170
	52		50 D0 00013	CALLS	#0, ALLOCATE_MEMORY	
	6E	2443424A	8F D0 00016	MOVL	R0, LKSB	: 2175
	50	04	AC D0 0001D	MOVL	#608387658, RESNAM	: 2176
04	AE		60 D0 00021	MOVL	SYSID, R0	
08	AE	04	A0 B0 00025	MOVL	(R0), RESNAM+4	
0C	AE		0A D0 0002A	MOVW	4(R0), RESNAM+8	: 2177
10	AE		6E 9E 0002E	MOVL	#10, RESNAM_DESC	: 2178
			7E 7C 00032	MOVAB	RESNAM, RESNAM_DESC+4	: 2189
			7E D4 00034	CLRQ	-(SP)	
			52 DD 00036	CLRL	-(SP)	
		0000V	CF 9F 00038	PUSHL	LKSB	
			7E D4 0003C	PUSHAB	ENTER_REMOTE_REQUEST_AST	
		24	AE 9F 0003E	CLRL	-(SP)	
7E		0208	8F 3C 00041	PUSHAB	RESNAM_DESC	
			52 DD 00046	MOVZWL	#520, -(SP)	
			05 DD 00048	PUSHL	LKSB	
			7E D4 0004A	PUSHL	#5	
				CLRL	-(SP)	

ASYNCHRON  
V04-002

Asynchronous service management

N 6  
15-Sep-1984 23:49:14  
14-Sep-1984 22:32:32

VAX-11 Bliss-32 V4.0-742  
[JOBCTL.SRC]ASYNCHRON.B32;3

Page 40  
(9)

00000000G	00	0B	FB	0004C	CALLS	#11, SYSS\$ENQ	:
	53	50	D0	00053	MOVL	R0, STATUS	:
	64	10	8A	00056	BICB2	#16, FLAGS	2202
00000689	8F	53	D1	00059	CMPL	STATUS, #1673	2203
		1A	12	00060	BNEQ	1\$	:
	64	10	88	00062	BISB2	#16, FLAGS	2206
		7E	7C	00065	CLRQ	-(SP)	2207
		7E	D4	00067	CLRL	-(SP)	:
		A2	DD	00069	PUSHL	4(LKSB)	:
00000000G	00	04	FB	0006C	CALLS	#4, SYSS\$DEQ	:
		52	DD	00073	PUSHL	LKSB	2208
00000000G	EF	01	FB	00075	CALLS	#1, DEALLOCATE_MEMORY	:
	11	53	E8	0007C	BLBS	STATUS, 2\$	2214
		53	DD	0007F	PUSHL	STATUS	2216
		7E	D4	00081	CLRL	-(SP)	:
		8F	DD	00083	PUSHL	#295954	:
00000000G	00	03	FB	00089	CALLS	#3, LIB\$SIGNAL	:
		04	00090	2\$:	RET		2217

; Routine Size: 145 bytes, Routine Base: CODE + 06C2

```
: 1187      2218 1 ROUTINE ENTER_REMOTE_REQUEST_AST(LKSB): NOVALUE=
: 1188      2219 1
: 1189      2220 1 ++
: 1190      2221 1
: 1191      2222 1 FUNCTIONAL DESCRIPTION:
: 1192      2223 1     This routine is the completion AST routine for obtaining another job
: 1193      2224 1     controller's remote request lock.
: 1194      2225 1
: 1195      2226 1 INPUT PARAMETERS:
: 1196      2227 1     LKSB           - Pointer to LKSB allocated from dynamic memory.
: 1197      2228 1
: 1198      2229 1 IMPLICIT INPUTS:
: 1199      2230 1     NONE
: 1200      2231 1
: 1201      2232 1 OUTPUT PARAMETERS:
: 1202      2233 1     NONE
: 1203      2234 1
: 1204      2235 1 IMPLICIT OUTPUTS:
: 1205      2236 1     NONE
: 1206      2237 1
: 1207      2238 1 ROUTINE VALUE:
: 1208      2239 1     NONE
: 1209      2240 1
: 1210      2241 1 SIDE EFFECTS:
: 1211      2242 1     NONE
: 1212      2243 1
: 1213      2244 1 --
: 1214      2245 1
: 1215      2246 2 BEGIN
: 1216      2247 2 MAP
: 1217      2248 2     LKSB:           REF BBLOCK;      ! Pointer to lock status block
: 1218      2249 2
: 1219      2250 2
: 1220      2251 2 ! Check status of the $ENQ.
: 1221      2252 2
: 1222      2253 2 IF NOT .LKSB[0,0,16,0]
: 1223      2254 2 THEN
: 1224      2255 2     SIGNAL(JBC$_COMREMJBC OR STS$K_ERROR, 0, .LKSB[0,0,16,0]);
: 1225      2256 2
: 1226      2257 2
: 1227      2258 2 ! Release the lock to enable the receiving job controller to recover it.
: 1228      2259 2
: 1229      2260 2 $DEQ(LKID=.LKSB[4,0,32,0]);
: 1230      2261 2
: 1231      2262 2
: 1232      2263 2 ! Deallocate the LKSB.
: 1233      2264 2
: 1234      2265 2 DEALLOCATE_MEMORY(.LKSB);
: 1235      2266 1 END;
```

```
0004 00000 ENTER_REMOTE_REQUEST_AST:
52      04 AC D0 00002      .WORD Save R2
                        MOVL  LKSB, R2
```

```
: 2218
: 2253
```

ASYNCHRON  
V04-002

Asynchronous service management

C 7  
15-Sep-1984 23:49:14  
14-Sep-1984 22:32:32

VAX-11 Bliss-32 V4.0-742  
[JOBCTL.SRC]ASYNCHRON.B32;3

Page 42  
(10)

	12	62	E8	00006	BLBS	(R2), 1\$	:	
	7E	62	3C	00009	MOVZWL	(R2), -(SP)	:	2255
		7E	D4	0000C	CLRL	-(SP)	:	
00000000G	00	8F	DD	0000E	PUSHL	#295954	:	
		03	FB	00014	CALLS	#3, LIB\$SIGNAL	:	
		7E	7C	0001B	CLRL	-(SP)	:	2260
		7E	D4	0001D	CLRL	-(SP)	:	
		A2	DD	0001F	PUSHL	4(R2)	:	
00000000G	00	04	FB	00022	CALLS	#4, SYSSDEQ	:	
		52	DD	00029	PUSHL	R2	:	2265
00000000G	EF	01	FB	0002B	CALLS	#1, DEALLOCATE_MEMORY	:	
		04	00032	RET			:	2266

; Routine Size: 51 bytes, Routine Base: CODE + 0753

```

: 1237 2267 1 GLOBAL ROUTINE QUEUE_MASTER_AST: NOVALUE=
: 1238 2268 1
: 1239 2269 1 ++
: 1240 2270 1
: 1241 2271 1 FUNCTIONAL DESCRIPTION:
: 1242 2272 1 This routine is the completion AST routine for the queue master lock.
: 1243 2273 1 It is entered when the queue master job controller fails and releases
: 1244 2274 1 the lock.
: 1245 2275 1
: 1246 2276 1 INPUT PARAMETERS:
: 1247 2277 1 Standard AST routine parameters (not used).
: 1248 2278 1
: 1249 2279 1 IMPLICIT INPUTS:
: 1250 2280 1 NONE
: 1251 2281 1
: 1252 2282 1 OUTPUT PARAMETERS:
: 1253 2283 1 NONE
: 1254 2284 1
: 1255 2285 1 IMPLICIT OUTPUTS:
: 1256 2286 1 NONE
: 1257 2287 1
: 1258 2288 1 ROUTINE VALUE:
: 1259 2289 1 NONE
: 1260 2290 1
: 1261 2291 1 SIDE EFFECTS:
: 1262 2292 1 NONE
: 1263 2293 1
: 1264 2294 1 --
: 1265 2295 1
: 1266 2296 2 BEGIN
: 1267 2297 2
: 1268 2298 2 Ensure that at least one job controller holds a timer on the timed job
: 1269 2299 2 queue.
: 1270 2300 2
: 1271 2301 2 AFTER_AST();
: 1272 2302 1 END;

```

00000000G EF 0000 0000  
00 FB 00002  
04 00009

.ENTRY QUEUE\_MASTER\_AST, Save nothing  
CALLS #0, AFTER\_AST  
RET

: 2267  
: 2301  
: 2302

; Routine Size: 10 bytes, Routine Base: CODE + 0786

ASYNCHRON  
V04-002

Asynchronous service management

E 7  
15-Sep-1984 23:49:14  
14-Sep-1984 22:32:32

VAX-11 Bliss-32 V4.0-742  
[JOBCTL.SRC]ASYNCHRON.B32;3

Page 44  
(12)

: 1274 2303 1 END  
: 1275 2304 0 ELUDOM

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
COMMON	5024	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, OVR, NOPIC, ALIGN(2)
CODE	1936	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	53	0	1000	00:01.4

: Information: 2  
: Warnings: 0  
: Errors: 0

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:ASYNCHRON/OBJ=OBJ\$:ASYNCHRON MSRC\$:ASYNCHRON/UPDATE=(ENH\$:ASYNCHRON)

: Size: 1886 code + 5074 data bytes  
: Run Time: 00:34.2  
: Elapsed Time: 03:54.5  
: Lines/CPU Min: 4040  
: Lexemes/CPU-Min: 39299  
: Memory Used: 380 pages  
: Compilation Complete

0191 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

BATCH  
LIS

BROADCAST  
LIS

BUFFERS  
LIS

CONTROL  
LIS

ASYNCHRON  
LIS

CHECKPROT  
LIS